Diplomarbeit

Entwicklung eines Tools zur Verwaltung und Generierung von XSL-Templates

Thorsten Lutz
WS 2003/04

Betreuung: Prof. Dr. Andreas Lux



Fachhochschule Trier
University of Applied Sciences
Fachbereich Design und Informatik
Studiengang Angewandte Informatik

In Zusammenarbeit mit



GE Fanuc Automation Europe S.A.
Zone Industrielle
L - 6468 Echternach

Grand Duché de Luxembourg

Ansprechpartner:
Dipl.-Wirtsch.-Ing. Danny Gießner



Fachhochschule Trier
University of Applied Sciences
Fachbereich Design und Informatik
Studiengang Angewandte Informatik

Autor:	Thorsten Lutz
Titel:	Entwicklung eines Tools zum Verwalten und Generieren von XSL-Templates
Studiengang:	Angewandte Informatik
Betreuung:	Prof. Dr. Andreas Lux

Es wird hiermit der Fachhochschule (University of Applied Sciences) die Erlaubnis erteilt, die Arbeit zu nicht-kommerziellen Zwecken zu verteilen und zu vervielfältigen.

Unterschrift des Autors

© Copyright Thorsten Lutz, 2003

Über diese Arbeit

Die vorliegende Diplomarbeit beschreibt eine Lösung zur Verminderung des Verwaltungsaufwandes für XSL-Templates, die zum Aufbau einer Webpräsenz auf Basis eines Content Management Systems benötigt werden am Beispiel des Intranet von GE Fanuc Automation Europe S.A.

Zunächst werden die verwendeten Kern-Technologien XML und XSL erläutert. Im Anschluss wird die Ausgangssituation analysiert. Dabei werden die verwendete Content Management Software, die Intranetstruktur und die vorhandenen XSL-Templates untersucht und Schwachpunkte beleuchtet.

Die gewonnenen Erkenntnisse bilden die Grundlage für eine Überarbeitung der XSL-Templates. Dazu wird ein neues Konzept erarbeitet und praktisch umgesetzt, bei dem die aufgetretenen Probleme berücksichtigt werden. Den Abschluss der Überarbeitung bildet ein Relaunch des Intranets.

Zur Verwaltung der überarbeiteten XSL-Templates wird ein Tool entwickelt. Es ist auf das erarbeitete Template-Konzept angepasst, kann jedoch auch für andere Webpräsenzen eingesetzt werden. Dies wird durch die Entwicklung von XSL-Templates für weitere Web-Auftritte auf Basis des Content Management Systems mit Hilfe des Tools praktisch umgesetzt.

Danksagung

An dieser Stelle möchte ich mich bei allen Personen bedanken, die mich während der Bearbeitungszeit mit wertvollen Hinweisen und Informationen unterstützt haben, sowie bei GE Fanuc Automation Europe S.A., die diese Arbeit ermöglicht haben.

Mein besonderer Dank gilt den Mitarbeitern des eBusiness-Teams Dipl.-Wirtsch.-Ing. Danny Gießner und Hans Jürgensen, sowie Prof. Dr. Andreas Lux (Fachhochschule Trier) für die hervorragende Betreuung während der Bearbeitung und der praktischen Umsetzung dieser Arbeit.

Thorsten Lutz, Echternach, im November 2003

Inhaltsverzeichnis

Abkürzungsverzeichnis

Bsp.	Beispiel
ca.	circa
CMS	Content Management System
CSS	Cascading Stylesheet
DOM	Document Object Model
GE	General Electrics
GML	Generalized Markup Language
HTML	Hypertext Markup Language
JS	JavaScript
JSP	Java Server Pages
LDAP	Lightweight Directory Access Protocol
o.V.	ohne Verfasserangabe
S.	Seite(n)
SGML	Standard Generalized Markup Language
URL	Uniform Resource Locator
Vgl.	Vergleiche, siehe
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language
XSLT	Extensible Stylesheet Language Transformation
z.B.	zum Beispiel

Abbildungsverzeichnis

Hinweis: Eingefügte Screenshots wurden teilweise nachbearbeitet, um unternehmens-interne Daten zu schützen.

Tabellenverzeichnis

1 Einleitung

1.1 Problembeschreibung

Das Unternehmen GE Fanuc Automation Europe S.A. betreibt ein internes Web (Intranet) auf Basis eines Content-Management-Systems (CMS). Um Informationen eines Content-Management-Systems als Webseiten darzustellen, benötigt man Schablonen (Templates), die die Informationen in das gewünschte Design „verpacken“.

Hauptaugenmerk wurde bei der Einführung des Systems auf die Akzeptanz durch die Benutzer und CMS-Redakteure gelegt. Es wurden Templates für verschiedenste Funktionalitäten und Darstellungsformen bereitgestellt. Wenn eine neue Funktion oder Darstellungsform der Informationen benötigt wurde, wurde ein neues Template erstellt. Ein schlüssiges Konzept für die Erstellung und Verwaltung der Templates gab es nicht.

Es entstand eine Vielzahl von Templates. Daraus resultieren 2 große Probleme.

1. Den CMS-Redakteuren, die für die Inhalte des Intranets verantwortlich sind, steht eine Vielzahl von Templates zur Auswahl, deren Namen nicht immer Aufschluss über die Funktion geben. Die Auswahl des geeigneten Templates ist schwierig.
2. Den CMS-Administratoren, die für die Funktion und Verwaltung des Content-Management-Systems verantwortlich sind, fällt es schwer, Änderungen durchzuführen, da nicht überschaubar ist welche Templates dafür zu ändern sind. Geplante Änderungen am Layout und eine Erweiterung der Funktionalität wurden aus diesem Grund lange aufgeschoben.

1.2 Ziel der Diplomarbeit

Ziel dieser Diplomarbeit ist es, die oben beschriebenen Probleme zu lösen. Dies wird in zwei Stufen geschehen.

Erstes Teilziel ist die Modularisierung der bestehenden XSL-Templates. Durch sinnvolles Zusammenfassen von Template-Funktionen wird die Anzahl der bestehenden XSL-Templates deutlich verringert und dadurch das Arbeiten mit den Templates für die CMS-Redakteure erleichtert. Dabei werden neue Funktionalitäten und das neue Design in die Templates eingebracht. Abschluss dieser Etappe wird der Relaunch[1] des Intranets mit den neuen Templates sein.

Die Entwicklung eines Tools zur Verwaltung und Generierung von XSL-Templates wird den CMS-Administratoren bei der Verwaltung bestehender Templates und der Erzeugung neuer Templates helfen. Das Tool wird in der Lage sein Templates für beliebige Web-Auftritte (Intranet, Internet, Extranet) zu verarbeiten. Bestehende Templates können gelesen und geändert werden. Zusätzlich wird es die Möglichkeit geben neue Templates zu erstellen, deren XSL-Code teilweise automatisch generiert wird.

[1] Relaunch: Wiedereinführung, Neuauflage

1.3 Vorgehensweise

Das folgende Grobkonzept zeigt, wie bei der Lösung der Aufgabe vorgegangen wird. Neben einer kurzen Beschreibung ist auch der geplante Zeitrahmen für die einzelnen Schritte angegeben. Für das gesamte Projekt ist ein Zeitrahmen von 19 Wochen vorgesehen.

1. **Einarbeitung in CMS-Technologie**
   Einarbeitung in Aufbau und Funktionsweise des unternehmensinternen Content Management Systems.
   Geschätzter Zeitaufwand: 5 Tage (bis 23.05.03)

2. **Einarbeitung in XML und XSL**
   Einarbeitung in die XML- und XSL-Thematik und das bei GE Fanuc verwendete Tool „XML Spy IDE". Dabei sollen (als Übung) auch kleinere Fehler in den aktuellen XSL-Templates des CMS analysiert und behoben werden.
   Geschätzter Zeitaufwand: 8 Tage (bis 06.06.03)

3. **Erstellung einer Übersicht über bestehende Templates und deren Verwendung**
   Alle momentan verwendeten Templates sollen erfasst werden. Zu jedem Template soll festgestellt werden, für welche Zwecke es verwendet wird.
   Geschätzter Zeitaufwand: 4 Tage (bis 13.06.03)

4. **Gesetzmäßigkeiten / Abhängigkeiten definieren**
   Die Verwendung der verschiedenen Templates unterliegt bestimmten „Regeln", die ein einheitliches Layout der Webseiten gewährleisten sollen. An bestimmten Positionen sind nur bestimmte Templates zulässig. Dies gilt dann natürlich auch für die einzelnen Module. Die Gesetzmäßigkeiten und Abhängigkeiten im Bezug auf die Verwendung der Module sollen definiert werden und in die Entwicklung des Tools einfließen.
   Geschätzter Zeitaufwand: 5 Tage (bis 20.06.03)

5. **Modularisierung der XSL-Templates**
   Die bestehenden Templates sollen sinnvoll in kleinere Module zerlegt werden. Mehrere Module bilden dann ein Template. Die einzelnen Module sollen die Basis für die Templates bilden, die später mit dem zu entwickelnden Tool erstellt werden können.
   Geschätzter Zeitaufwand: 15 Tage (bis 11.07.03)

6. **Übersicht verfügbarer Templates für CMS-Redakteure erstellen**
   Für die Redakteure, die das Content Management System pflegen, soll eine Online-Übersicht über die verfügbaren Templates erstellt werden. Diese soll Informationen über die Eigenschaften des Templates und Informationen über die Verwendungsmöglichkeiten geben. Eine Vorschau für jedes Template soll ebenfalls in der Übersicht enthalten sein.
   Geschätzter Zeitaufwand: 5 Tage (bis 18.07.03)

7. **Entwicklung eines Tools zur Generierung neuer XSL-Templates**
   Es soll ein Tool entwickelt werden, mit dem es möglich ist aus einer Anzahl existierender Module neue Templates zu generieren. Dabei sollen die vorher aufgestellten Gesetzmäßigkeiten und Abhängigkeiten berücksichtigt werden. Das Tool soll anhand dieser Regeln und dem Verwendungszweck des neuen Templates bestimmen, welche Module eingesetzt und wie diese miteinander kombiniert werden können.

Geschätzter Zeitaufwand:	Konzeption	15 Tage (bis 15.08.03)
	Implementierung	20 Tage (bis 12.09.03)
	Test / Analyse	10 Tage (bis 26.09.03)

1.4 GE Fanuc Automation Europe S.A.

1.4.1 Vorstellung des Unternehmens

Das Unternehmen GE Fanuc Automation wurde 1986 gegründet und ist ein Joint Venture von General Electrics (GE) und FANUC Ltd.
General Electrics hat seinen Hauptsitz in den USA und ist mit weltweit rund 300.000 Mitarbeitern einer der größten Konzerne der Welt.
Das japanische Unternehmen FANUC Ltd. gilt als Pionier-Unternehmen mit mehr als 30 Jahren Erfahrung im Bereich Industrie-Automation und ist weltweiter Marktführer auf dem Gebiet der CNC-Technologie.

GE Fanuc Automation entwickelt Produkte und Lösungen für den weltweiten Industrie-Automationsmarkt. GE Fanuc Automation Europe, mit Hauptsitz in Echternach, Luxemburg, betreut Kunden in Europa, Afrika, sowie im nahen und mittleren Osten und hat über 400 Mitarbeiter in Luxemburg und den nationalen Niederlassungen in Großbritannien, Frankreich, Italien, Spanien, Polen, Deutschland, Schweiz, Niederlande, Tschechien und in Skandinavien.
Der größte Konkurrent für GE Fanuc Automation Europe ist das deutsche Unternehmen Siemens.

General Electrics betreibt ein eigenes weltweites Computernetzwerk, an das auch GE Fanuc Automation Europe angeschlossen ist.

1.4.2 Übersicht über die GE Fanuc Webs

Das Unternehmen GE Fanuc betreibt seit fast 10 Jahren ein unternehmensinternes Web (Intranet). Alle wichtigen Informationen und aktuelle Daten der einzelnen Abteilungen werden den Mitarbeitern über das Intranet zur Verfügung gestellt. Dazu zählen unter anderem Produktinformationen, Projektübersichten, Telefonlisten, aktuelle Statistiken und Kalkulationen.
Das Intranet basierte zunächst auf statischen HTML-Seiten. Eine stetige Zunahme der Webseiten führte zu einem übermäßig hohen Administrationsaufwand.

Neben dem Intranet betreibt GE Fanuc Automation Europe 15 statische Internet-Webs für die als wichtig eingestuften Länder (in Landessprache) sowie einen statischen Internet-Auftritt für Europa. Auf diese Webs trifft die gleiche Problematik zu.

Aus diesem Grund entschloss man sich für die Einführung eines Content Management Systems (CMS), wodurch der Administrationsaufwand für die Webs wieder gesenkt werden sollte. Da große Teile der Internet-Inhalte aus Seiten des Intranet übernommen werden, wurde als erstes das Intranet im Mai 2002 auf CMS umgestellt

Das Intranet stellt abteilungsbezogen Informationen bereit. Für den Inhalt und die Struktur sind die CMS-Redakteure der einzelnen Abteilungen verantwortlich. Für die Pflege und Funktion des Intranets und des CMS sind CMS-Administratoren, bei GE Fanuc Automation S.A. die Mitarbeiter des eBusiness-Teams, verantwortlich. Dazu gehören auch das Layout und verschiedene Tools, die im Intranet bereitgestellt werden.

Seit der CMS-Einführung im Mai 2002 ist das Intranet stetig gewachsen. Mit der Anzahl der Medien, Contents und Navigationspunkte stiegen auch die Anforderungen im Bezug auf Tools und Darstellungsformen.

Nach und nach sollen die Internet-Auftritte ebenfalls in das CMS integriert werden.
Am Ende werden insgesamt 22 Webs durch das CMS verwaltet werden.

2 Grundlagen und Abgrenzungen

2.1 XML und XSL

2.1.1 Markup-Sprachen

Markup-Sprachen (engl. „Auszeichnungssprachen“) beschreiben die logischen Teile von Dokumenten. Markup-Sprachen sollen die Struktur eines Dokuments beschreiben und von einem Programm oder einem Menschen eindeutig interpretierbar sein. Sie grenzen sich von Programmiersprachen ab, da sie nicht über deren typischen Konstrukte Schleifen, Verzweigungen, Funktionen und Variablen verfügen.
Man unterscheidet zwei Arten der Auszeichnungen. **Textauszeichnungen** dienen dazu, die Bedeutung einer Dokumentpassage zu beschreiben (z.B. Überschrift), während **Layoutauszeichnungen** die Darstellung einer Passage beschreiben (z.B. Fettdruck). Die Auszeichnung von Textpassagen geschieht typischerweise mit Hilfe von Tags (Marken), die den entsprechenden Text einschließen.

1969 erfanden Charles Goldfarb, Edward Mosher und Raymond Lorie mit der Generalized Markup Language (GML) eine Metasprache, mit der neue Markupsprachen generiert werden konnten. 1986 wurde SGML (Standard Generalized Markup Language) zum ISO-Standard 8879. SGML stellt eine Weiterentwicklung von GML dar und bietet eine völlige Plattform-Unabhängigkeit und die Möglichkeit der Daten-Publizierung auf verschiedensten Medien. Die wohl bekannteste SGML-Anwendung ist HTML.[2]

Da der SGML-Standard sehr komplex ist, begann das W3C-Konsortium 1996 mit der Arbeit an XML[3] (Extensible Markup Language), die SGML erweitern bzw. vereinfachen sollte. „Die Extensible Markup Language wurde vom W3C als "SGML for the Web" konzipiert. XML stellt eine Auszeichnungssprache dar, die die wesentlichen Freiheiten von SGML teilt, ohne die Kompliziertheit des ISO-Standards zu besitzen.“[4]

2.1.2 XML

XML ist eine vereinfachte Form von SGML (Standard Generalized Markup Language), mit der man eigene Markup-Sprachen definieren kann. Hauptsächlich dient XML der Beschreibung, der Haltung und dem Transfer von Daten.
Die mit XML definierten Markup-Sprachen werden als XML-Anwendungen bezeichnet. Die einzelnen Elemente der XML-Anwendung, ausgezeichnet durch Start-Tag und End-Tag, bilden Knoten (Nodes), durch deren Verschachtelung eine Baumstruktur entsteht. Die Syntax, Struktur und Bedeutung der Tags wird für jede XML-Anwendung mit einer DTD (Document Type Definition) oder einem Schema definiert. Die Verarbeitung von XML-Anwendungen erfolgt mit Hilfe eines XML-Parsers.

---

[2] Vgl. Geschichte des Hypertextes; Online im Internet; URL: http://ddi.uni-paderborn.de/didaktik/lehre/sose2000/tide/html_geschichte.html (13.11.2003)
[3] Online im Internet; URL: http://www.w3.org/XML/ (13.11.2003)
[4] Quelle: Internetbüro Literaturwissenschaft Fakultät für Linguistik und Literaturwissenschaft URL: http://www.uni-bielefeld.de/lili/organisationen/intbuero/hilfen/glossar/xml.html (13.11.2003)

XML kennt nur Textauszeichnungen. Innerhalb einer XML-Anwendung können keine Angaben darüber gemacht werden, wie bestimmte Textpassagen darzustellen sind. Zum Festlegen der Darstellung von XML-Daten dienen XSL- oder CSS[5]-Dateien.

Beispiel einer XML-Datei:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<Fahrzeuge>
  <PKW Marke="Opel" Modell="Astra">
    <Farbe>weiss</Farbe>
    <Baujahr>1995</Baujahr>
  </PKW>
  <PKW Marke="VW" Modell="Golf">
    <Farbe>rot</Farbe>
    <Baujahr>1997</Baujahr>
  </PKW>
  <LKW Marke="Mercedes" Modell="Sprinter">
    <Farbe>grau</Farbe>
    <Baujahr>1996</Baujahr>
  </LKW>
</Fahrzeuge>
```

2.1.3 XSL

XSL steht für „Extensible Stylesheet Language“ und ist eine vom W3C-Konsortium standardisierte Sprachfamilie zur Verarbeitung von XML-Dokumenten. XSL selbst basiert auf einem XML-Vokabular. Eine XSL-Anwendung bezeichnet man als XSL-Stylesheet oder Template. Es beschreibt die Darstellung einer XML-Anwendung. Die strukturellen und semantischen Auszeichnungen der XML-Anwendung werden in layout-spezifische Auszeichnungen transformiert.

XSL besteht aus 3 Teilen:

- **XSL-FO**[6] („Formatting Objects“) ist eine Formatierungssprache und dient zur Beschreibung von Layouts. XSL-FO kann nicht direkt auf Daten eines XML-Dokuments zugreifen. Da XSL-FO für diese Arbeit nicht relevant ist wird es nicht näher erläutert.
- **XSL-Transformations**[7] (XSLT) bildet die Transformationssprache, mit der die Informationen eines XML-Dokuments in eine Darstellungsform überführt werden. Dabei werden Regeln angewendet, die in einem XSL-Stylesheet festgelegt werden.
- **XPath**[8] bietet zahlreiche Funktionen zum Selektieren von Elementen und zum Abfragen bestimmter Element-Eigenschaften (z.B. Textlänge). XPath leitet sich von XPointer[9], einer vom W3C entwickelten Sprache zur Verlinkung von Dokumenten ab.

---

[5] CSS: **C**ascading **S**tyle**S**heets; Online im Internet; URL: http://www.w3.org/Style/CSS/ (13.11.2003)
[6] Online im Internet; URL: http://www.w3.org/TR/xsl/ (13.11.2003)
[7] Online im Internet; URL: http://www.w3.org/TR/xslt/ (13.11.2003)
[8] Online im Internet; URL: http://www.w3.org/TR/xpath/ (13.11.2003)
[9] Online im Internet; URL: http://www.w3.org/TR/xptr/ (13.11.2003)

Um eine XML-Anwendung mit Hilfe von XSL-Stylesheets in ein anderes Format umzuwandeln, benötigt man einen XSLT-Prozessor. Der XSLT-Prozessor ist eine Anwendung, die ein XSL-Stylesheet auf einen XML-Baum anwendet und daraus einen Ergebnisbaum (z.B. ein HTML-Dokument) erzeugt. Ein XSL-Stylesheet kann im Normalfall nur auf eine bestimmte Klasse von XML-Dokumenten angewendet werden, da es dessen Elemente „kennen" muss.

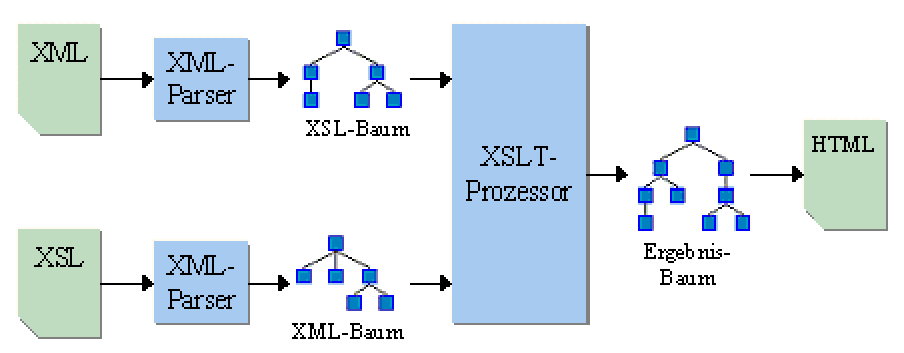


Abbildung 2-1: Funktionsweise des XSLT-Prozessors
Quelle: eigene Darstellung

Elemente eines Templates

Eine Beschreibung aller XSL-Tags, ihrer Attribute und Funktionsweise würde an dieser Stelle zu weit führen. Daher werden nur die für diese Arbeit wichtigen XSL-Tags und deren wichtigste Attribute kurz erläutert.

Da in vielen Tags die Attribute „select" und „test" verwendet werden, werden sie hier kurz beschrieben. Über das „select"-Attribut wird ein Element selektiert. Dies kann ein Knoten des XML-Dokuments oder ein im Template definierter Parameter sein. Der Wert des Attributs muss ein XPath-Ausdruck sein. „test" dient zum Vergleichen des Ergebnisses zweier XPath-Funktionen. Dieses Attribut findet Anwendung bei Verzweigungen.

Basis des XPath-Ausdrucks ist immer das aktuell selektierte Element.

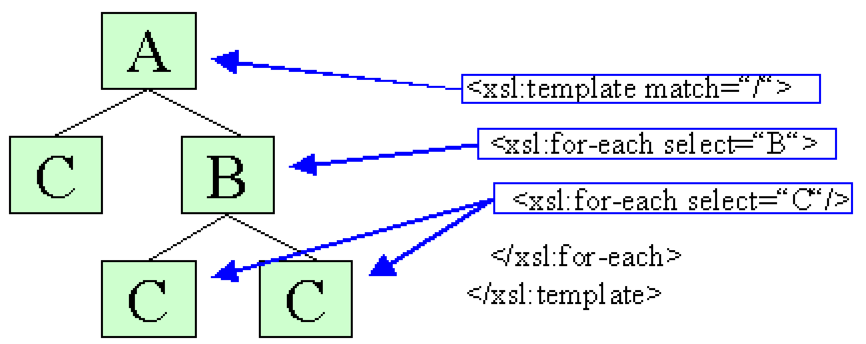


Abbildung 2-2: Selektierung mittels XPath-Anweisung
Quelle: eigene Darstellung

Da ein XSL-Stylesheet eine XML-Anwendung darstellt, beginnt jedes Template mit der XML-Deklaration („`<?xml version="1.0"?>`").

- **Das Wurzelelement**
  `<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/ XSL/Transform">`
  Dieser Tag bildet das Wurzelelement (Dokumentelement) des Templates. In ihm werden die im Template verwendeten XML-Namensräume (hier „xsl") definiert. Die Deklaration des Namensraums für XSL ist zwingend, um ein funktionsfähiges

Template zu erstellen. Durch den Namespace-Präfix (hier „xsl“) erkennt der XSLT-Prozessor bei der Verarbeitung des Dokuments die XSL-Anweisungen.

- **Importanweisung**
  ```
  <xsl:import href="ictdb://3de70d6b-2f7f-b1fa-168a-018cc0298603"/>
  ```
  XSL bietet die Möglichkeit Templates zu modularisieren, d.h. es ist möglich Templates in andere Templates einzufügen. Template-Regeln, die von mehreren Templates verwendet werden, können in einem zentralen XSL-Dokument abgelegt werden. Dazu dient die Anweisung „import“ . Importanweisungen müssen die ersten Kindknoten des Wurzelelements sein. Das Attribut „href“ gibt die Quelle des zu importierenden Templates (hier ein CMS-Datenbank-Inhalt) an. Die Kindknoten des Wurzelelements des zu importierenden Templates werden an die Stelle der Import-Anweisung in den XSL-Baum eingefügt.
- **Ausgabeformat festlegen**
  ```
  <xsl:output method="html" version="4.0" encoding="ISO-8859-1"/>
  ```
  Die Output-Anweisung bestimmt das Ausgabeformat des durch die Transformation entstehenden Ergebnisbaumes. Es gibt drei Standard-Ausgabeformate, „xml“, „html“ und „text“, die im Attribut „method“ festgelegt werden. Es besteht die Möglichkeit neue Ausgabeformate zu definieren. Weitere Attribute erlauben je nach Ausgabeformat speziellere Einstellungen (hier HTML-Version und Codierung) für die Ausgabe. Dieser Knoten muss direkter Kindknoten des Wurzelelements sein und vor der ersten Template-Regel stehen.
- **Parameter**
  ```
  <xsl:param name="Content">...
  ```
  Mit Hilfe dieser Anweisung kann man Parameter setzen. Der Name des Parameters wird im Attribut „name“ festgelegt. Der Wert des Parameters ist der Inhalt des Tags (Kindknoten). Der Wert kann eine Zeichenkette oder ein Konstrukt aus XSL-Anweisungen sein. Als direkte Kindknoten des Wurzelelements, können Parameter im gesamten Template angesprochen werden, werden sie innerhalb einer Template-Regel verwendet, sind sie nur innerhalb dieser Regel gültig. Parameterwerte lassen sich nach ihrer Deklaration nicht mehr ändern. Für einen in einer Template-Regel deklarierten Parameters kann beim Aufruf dieser Template-Regel aus einer anderen ein Parameterwert „übergeben“ werden, der den Wert der Deklaration überschreibt. Innerhalb eines XPath-Ausdrucks wird ein Parameter mit vorangestelltem „$“ angesprochen (z.B. <xsl:value-of select=„$Content“/>).
- **Template-Regeln**
  ```
  <xsl:template name="master" match="/">...
  ```
  Eine Template-Regel (Template-Rule) enthält Anweisungen, die festlegen wie bestimmte Teile des XML-Baums umgewandelt werden sollen. Über das Attribut „name“ kann eine Regel gezielt aufgerufen werden. Das Attribut „match“ bestimmt, auf welchen Knoten (hier Wurzel des XML-Dokuments) eine Regel angewandt wird.
- **Schleifen**
  ```
  <xsl:for-each select="PKW"> …
  ```
  Die im Tag eingeschlossenen Knoten werden auf jedes Element angewendet, auf das das select-Attribut (hier alle PKW-Elemente, die direktes Kind des aktuellen Elements sind) zutrifft.

- **Verzweigungen**
  In XSL gibt es 2 Arten der Verzweigung:
  1.
```
<xsl:if test="count(PKW) &gt; 0">…
```
  Bei der If-Anweisung werden die eingeschlossenen Knoten angewandt, wenn die Bedingung (hier: Anzahl PKW-Elemente größer 0), die über das Attribut „test“ festgelegt wird, erfüllt ist. Ansonsten wird der Tag übersprungen.
  2.
```
<xsl:choose>
     <xsl:when test="count(PKW) &gt; 0"> … </xsl:when>
     …
     <xsl:otherwise> … </xsl:otherwise>
<xsl:choose>
```
  Eine erweiterte Form der Verzweigung stellt das Choose-Konstrukt dar. Innerhalb des Choose-Tags können beliebig viele When-Anweisungen, die wie If-Anweisungen arbeiten, stehen. Es wird der erste Teilbaum durchlaufen, bei dem die jeweilige Bedingung wahr ist. Zusätzlich besteht die Möglichkeit als letzten Kindknoten der Choose-Anweisung eine Otherwise-Anweisung einzufügen. Sie wird ausgeführt, wenn alle vorherigen When-Bedingungen falsch sind.
- **Wert aus dem XML-Dokument auslesen**
```
<xsl:value-of select="Farbe"/>
```
  Zum Auslesen von Werten aus einem XML-Dokument verwendet man die Value-Of-Anweisung. Der Wert des im Attribut „select“ angegebenen Elements (hier Farbe) wird in den Ergebnisbaum geschrieben. Existieren mehrere Elemente dieses Namens, wird das erste ausgewählt.
- **Attribute erzeugen**
```
<xsl:attribute name="src">…
```
  Diese Anweisung erlaubt das dynamische Erzeugen von Attributen. Über das Attribut „name“ wird der Name des Attributes festgelegt. Der zugewiesene Wert kann innerhalb des Tags festgelegt werden oder über das Attribut „select“ angegeben werden.
- **Aufruf von Template-Regeln**
```
<xsl:call-template name="tabelle"> ...
```
  Die Call-Template-Anweisung dient zum Aufrufen einer Template-Regel. Der Name der aufzurufenden Template-Regel wird im Attribut „name“ festgelegt. Innerhalb dieses Tags kann man mit Hilfe der With-Param-Anweisung (<xsl:with-param name="Content"> ...) kann man Parameter-Werte an die aufgerufene Template-Regel übergeben. Das Attribut „name“ bestimmt den zu überschreibenden Parameter. Dieser Parameter muss in der aufgerufenen Template-Regel deklariert sein. Den Wert kann über das Attribut „value“ oder als Kindknoten des With-Param-Knotens angegeben werden.

Beispiel eines XSL-Templates zum Einsatz mit dem XML-Beispiel (siehe Kapitel 2.1.2).

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:output method="html"/>

  <xsl:template name="liste_fahrzeuge">
    <xsl:if test="position() = 1">
      <xsl:call-template name="tabellenkopf"/>
    </xsl:if>
    <tr>
```

```
        <td><xsl:value-of select="@Modell"/></td>
        <td><xsl:value-of select="Farbe"/></td>
        <td><xsl:value-of select="Baujahr"/></td>
      </tr>
    </xsl:template>

    <xsl:template name="tabellenkopf">
      <tr>
        <td class="head">Modell</td>
        <td class="head">Farbe</td>
        <td class="head">Baujahr</td>
      </tr>
    </xsl:template>

    <xsl:template match="/">
      <HTML>
        <HEAD>
          <TITLE>XSL-Beispiel</TITLE>
          <style type="text/css">
            td {border: 1px solid #003333; text-align: center}
            .head {background-color: #006666; width: 100px;}
          </style>
        </HEAD>
        <BODY>
          <p>PKW's
            <table  cellpadding="0"  cellspacing="0"  border="0">

              <xsl:for-each select="Fahrzeuge/PKW">
                <xsl:call-template name="liste_fahrzeuge"/>
              </xsl:for-each>
            </table>
          </p>
              <p>Es gibt <xsl:value-of
      select="count(Fahrzeuge/LKW)"/> LKW's und
            <xsl:value-of select="count(Fahrzeuge/Schiff)"/>
            Schiffe.
          </p>
        </BODY>
      </HTML>
    </xsl:template>
  </xsl:stylesheet>
```

Funktionsweise des XSLT-Prozessors

Zunächst wird das XSL-Template eingelesen. Es wird beim Wurzelelement (`<xsl:stylesheet>`) begonnen. Anschließend werden die Importanweisungen ausgeführt und die importierten Templates in die Baumstruktur „eingehangen“. Dann folgt die Output-Anweisung, danach globale Parameter und Template-Regeln.
Anschließend beginnt die Verarbeitung des XML-Dokuments. Der XSLT-Prozessor durchsucht das XML-Dokument bei der Wurzel beginnend nach einem Knoten für den eine Template-Regel existiert und führt diese auf dem gefundenen Element aus. Eine Template-Regel besagt: „Gebe alles aus, was innerhalb dieser Regel steht, und führe alle Anweisungen mit xsl-Namensraum aus“.
Anschließend wird die Suche fortgesetzt und ggf. eine weitere Template-Regel auf diese Weise ausgeführt.
Nach Abarbeiten des gesamten XML-Dokuments wird der Ergebnisbaum in das angegebene Ausgabeformat umgewandelt und ausgegeben.

Durch das Importieren von Templates kann die Situation entstehen, dass mehrere Template-Regeln mit gleichem Namen existieren. Dann wird die Template-Regel aufgerufen, die als erste (zeitlich) im XSL-Baum existierte.
Eine weitere Konfliktsituation ergibt sich, wenn auf ein XML-Element mehrere Template-Regeln „matchen“[10]. Die Verhaltensweisen der XSLT-Prozessoren in diesem Fall sind nicht einheitlich. Über zusätzliche Attribute („priority“ und „modes“) in der Template-Anweisung kann die Verhaltensweise des Prozessors in diesem Fall beeinflusst werden.

Die verbreitetesten XSLT-Prozessoren sind die Open-Source-Programme „Xalan“, „Saxon“ und das Microsoft-Produkt „msxml“.

Konzepte der Transformation für Webseiten
Zur Transformation der XML-Anwendung in ein HTML-Dokument gibt es grundsätzlich 3 Ansätze.
Der erste Ansatz ist, die XML-Daten bereits vor der Bereitstellung auf dem Server zu transformieren und als HTML-Dokument auf dem Server abzulegen. Der Vorteil liegt in der garantierten Übertragung von HTML-Daten. Auf Fehler kann bereits vor der Bereitstellung auf dem Server reagiert werden. Der Nachteil liegt darin, dass durch diese Methode die Funktionsvielfalt von XSL stark eingeschränkt wird.
Die zweite Variante ist das Ablegen von XML-Daten und XSL-Stylesheets auf dem Server. Die Transformation findet bei Bedarf auf dem Server statt und der ausgegebene HTML-Code wird gesendet. Dies ist die zurzeit beste Variante, da die volle XSL-Funktionalität genutzt werden kann und die Darstellung dennoch auf jedem Browser garantiert ist, da reiner HTML-Code gesendet wird. Mögliche Transformationsfehler treten unmittelbar beim Aufrufen der Webseite auf. Auf diese Fehler kann nicht reagiert werden. Sie werden möglicherweise für einen Kunden sichtbar.
Die dritte Möglichkeit besteht darin, XML-Dokument und XSL-Stylesheet vom Server zum Browser zu senden und die Transformation vom Browser ausführen zu lassen. Die meisten Browser der neuesten Generation verfügen über einen integrierten XSLT-Prozessor. Ein Großteil der Benutzer verfügt jedoch noch nicht über einen solchen Browser, daher ist dieser Ansatz derzeit nicht zu empfehlen.

Dies ist der mit Hilfe des XSL-Stylesheets erzeugte Ergebnisbaum. Es handelt sich um eine reine HTML-Datei.

```
<HTML>
  <HEAD>
    <TITLE>XSL-Beispiel</TITLE>
    <style type="text/css">
      td {border: 1px solid #003333; text-align: center}
      .head {background-color: #006666; width: 100px;}
    </style>
  </HEAD>
  <BODY>
    <p>PKW's
      <table cellpadding="0" cellspacing="0" border="0">
      <tr>
          <td class="head">Modell</td>
          <td class="head">Farbe</td>
```

[10] „matchen“: das Attribut „match“ der Template-Regel trifft auf das selektierte XML-Element zu

```
              <td class="head">Baujahr</td>
            </tr>
            <tr>
              <td>Astra</td>
              <td>weiss</td>
              <td>1995</td>
            </tr>
            <tr>
              <td>Golf</td>
              <td>rot</td>
              <td>1997</td>
            </tr>
          </table>
        </p>
        <p>Es gibt 1 LKW's
          und 0 Schiffe.
        </p>
      </BODY>
    </HTML>
```

PKW's

Modell	Farbe	Baujahr
Astra	weiss	1995
Golf	rot	1997

Es gibt 1 LKW's und 0 Schiffe.

Abbildung 2-3: HTML-Ausgabe nach der XSL-Transformation
Quelle: Screenshot der Browserausgabe des Beispiels

Vorteile und Einsatz von XSL
Durch den Einsatz von XML und XSL erhält man eine Trennung von Daten und deren Darstellungsform.
Auf eine XML-Datei können verschiedene Templates angewendet werden. So lassen sich XML-Daten plattform-spezifisch darstellen (z.B. WWW, WAP-Handy, PDA). Im Cross-Media-Publishing-Bereich, bei dem unterschiedliche Plattformen „bedient" werden müssen, ist dies der Grund für den häufigen Einsatz von XML/XSL.
Die Möglichkeit, ein Template auf mehrere XML-Dokumente anzuwenden, macht XML/XSL für den Content-Management-Bereich interessant. Verschiedenste Daten können mit dem gleichen Layout dargestellt werden. Der Autor des XML-Dokuments muss sich nicht mit Layout-Fragen befassen. Die Verantwortung für Datenpflege und Datenausgabe kann daher bei verschiedenen Personen liegen. Eine Layout-Änderung kann vorgenommen werden ohne die Gefahr eines versehentlichen Datenverlustes, da die Daten von den Änderungen „unberührt" bleiben.

2.2 Content Management Systeme

2.2.1 Begriffsklärung

Content Management Systeme (CMS) werden eingesetzt, um eine umfangreiche Sammlung von verschiedensten Dokumenten flexibel und mit angemessenem Aufwand verwalten zu können.

Der Begriff Content steht dabei für alle Information bzw. Dokumente, die sich in digitaler Form publizieren lassen (Texte, Bilder, Videos, ...). Management umschreibt die Erstellung, Bearbeitung, Verwaltung und Archivierung dieser Inhalte. Das System bildet die technische Basis, meist eine datenbankbasierte Anwendung, die aus verschiedenen Komponenten besteht.

Ein CMS-System, das speziell für die Verwaltung von Webseiten gedacht ist wird auch Web Content Management System (WCMS) genannt. Die Grenzen zwischen CMS und WCMS sind nicht eindeutig.

2.2.2 Merkmale eines CMS

1. Trennung von Inhalt und Layout

Bei Content Management Systemen gibt es eine klare Trennung zwischen Inhalten (Daten) und deren Darstellung. Eine Webseite wird durch das CMS erzeugt, indem Contents in ein vorgegebenes Seitengerüst (Template) eingefügt werden.

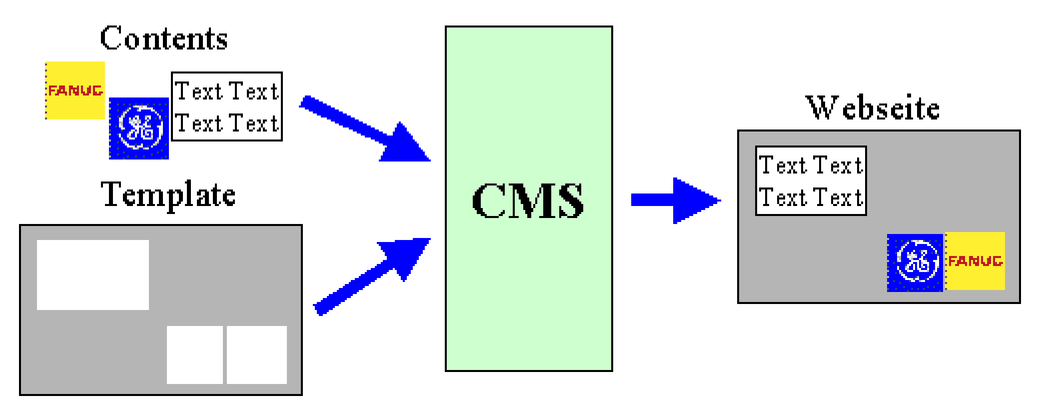


Abbildung 2-4: CMS-Webseitenaufbau
Quelle: eigene Darstellung

Durch Verwendung verschiedener Templates entsteht die Möglichkeit CMS-Inhalte für verschiedenste Anwendungen (z.B. PDA, WAP) zu nutzen. Durch diese Systematik ist ein unkomplizierter plattformübergreifender Content-Austausch möglich.

2. Proportionales Wachstum von Inhalt und Aufwand

Bei steigender Anzahl von Webseiten, steigt beim herkömmlichen Web-Publishing der Verwaltungsaufwand überproportional. Bei Verwendung eines WCMS ergibt sich ein proportionaler Anstieg von Wachstum und Aufwand. CMS-Systeme sind aus diesem Grund besonders für umfangreiche Webseiten interessant.

3. Trennung von Funktionen nach Rechten (Sicherheit, Qualität)

Bei CMS können beliebige Nutzer, Gruppen, Rollen oder auch Gruppenrollen mit ganz bestimmten Rechten versehen werden. Die zugewiesenen Rechte richten sich nach der jeweiligen Aufgabe.

4. Dezentralisierung der Webseitenbearbeitung

Durch die Trennung von Inhalt und Layout ist es möglich, auch ohne Programmierkenntnisse Webinhalte zu erzeugen oder zu ändern. Theoretisch könnte jeder Mitarbeiter an der Webseiten-Bearbeitung beteiligt sein. In der Regel gibt es einen ausgewählten Personenkreis, der für die Webseitengestaltung zuständig ist. Durch die Rechtevergabe kann genau gesteuert werden, wer welche Inhalte ändern darf.

Zum Erstellen und Publizieren wird kein Fachpersonal benötigt. Da die Inhalte direkt vom Verfasser publiziert werden verkürzt sich die „Time To Web“[11].

5. Flexible Änderungen am Layout

Ein Redesign der Webseiten wird wesentlich vereinfacht, weil nicht jede Webseite erneuert werden muss. Es ist nur eine Änderung der bzw. eines Templates nötig. Weil ein Template im Normalfall für mehrere Webseiten genutzt wird, entsteht eine erhebliche Zeitersparnis. Dabei muss nicht auf den eigentlichen Content geachtet werden.

6. Meta-Informationen und Attribute

Durch die datenbankbasierte Verwaltung der Contents ist es möglich, für alle Contents Meta-Informationen (z.B. Angaben über Autor, Erstellungsdatum, ...) anzulegen. Aufgrund dieser Angaben ist es möglich einzelne Contents gezielt nach bestimmten Kriterien auszuwählen.

Viele CMS-Systeme bieten die Möglichkeit zusätzliche frei definierbare Meta-Informationen (Attribute) anzulegen.[12]

[11] Time To Web: Zeit bis eine Information im Web erscheint.

[12] Vgl. Büchner, Traub, Zahradka, Zschau; S.106

3 Analyse der Ausgangssituation

3.1 Das eingesetzte CMS-System IC Content 4.0

GE Fanuc setzt die Content Management Software „IC Content 4.0“ der Firma IC Team GmbH aus Trier ein. IC Content basiert auf einer Client-Server-Architektur. Als Client dient ein Browser. Als Server dient eine Java-Applikation, die auf einem BEA WebLogic[13]-System oder einem jBoss-Application-Server[14] arbeitet. Während das Produktivsystem noch auf BEA-Basis arbeitet, wurden das Testsystem und die Webausgabe bereits auf jBoss umgestellt. Die Unterschiede zwischen den beiden Systemen sind für die CMS-Administratoren nur in Form einer leicht unterschiedlichen Ordnerstruktur sichtbar und werden hier nicht näher erläutert. Zur Datenhaltung dient eine mySQL-Datenbank. IC Content erfüllt alle Merkmale eines CMS (siehe Kapitel 2.2.2) und bietet zusätzliche Funktionalitäten.

Das Assetmanagement[15] von IC Content trennt nicht nur Inhalt und Darstellung. Die Inhalte werden in Content und Medien unterteilt. Für die Darstellung sind Templates verantwortlich. Zusätzlich werden Navigationspunkte als eigenständige Elemente behandelt.

1. Content ist ein CMS internes Objekt, das den eigentlichen Inhalt, der in HTML-Form vorliegt, enthält. Einem Content können Medien zugeordnet werden.
2. Ein Medium kann ein beliebiges Dokument in Dateiform sein (in der Regel Bilder, Excel-Tabellen, Word- und PDF-Dokumente).
3. Templates sind XSL-Stylesheets. Sie bestimmen die Darstellung der ausgegebenen HTML-Seiten.
4. Navigationspunkte dienen zum Aufbau der Navigationsstruktur. Ihnen können Templates, Contents und Medien zugeordnet werden, aus denen die HTML-Seite gebildet wird.

Zu diesen vier Elementen legt das CMS zusätzliche Meta-Informationen an, die durch selbstdefinierte Attribute erweitert werden können. Die selbstdefinierten Attribute werden in der Attribut-Administration verwaltet. Mehrere Attribute können zu einer Attribut-Klasse zusammengefasst werden. Die wichtigsten Attribut-Klassen für diese Arbeit sind „Author Information“ und „ExLink“. Die Klasse „Author-Information“ beinhaltet die Attribute „Author“, „Email“ und „CreateDate“. Diese Informationen dienen dazu, den Autor eines Contents und das Erstellungsdatum anzuzeigen. „ExLink“ enthält nur ein Attribut „exlink“. Über dieses Attribut wird die URL zu einem Medium angegeben, das nicht im CMS verwaltet wird, aber dennoch in der CMS-basierten Webseite angezeigt werden soll.

Um einen Content oder ein Medium im Intranet „sichtbar“ zu machen, muss der Content bzw. das Medium einem Navigationspunkt zugeordnet werden. Man spricht vom Anhängen eines Contents. Dem Navigationspunkt muss ein Template zugewiesen werden. Alle Elemente müssen auf der vorgesehenen Publikationsstufe publiziert sein.

---

13 Online im Internet; URL: http://de.bea.com/produkte/weblogic_server_81.jsp (13.11.2003)

14 Online im Internet; URL: http://www.jboss.org/index.html?op=userdisplay&id=developers/projects/jboss (13.11.2003)

15 Asset: (engl: Wert, Schatz) Im Zusammenhang mit CMS versteht man unter Assets alle Medien und Dokumente.

Es existieren mehrere Publikationsstufen. Dies dient der Qualitätssicherung. Die CMS-Redakteure haben die Möglichkeit ihre Änderungen zu prüfen, oder prüfen zu lassen, bevor sie für andere Benutzer sichtbar werden.
Zum sicheren Arbeiten mit einem Element gibt es eine CheckOut/CheckIn-Technik, die verhindert, dass mehrere Personen gleichzeitig ein Element bearbeiten.

ICContent stellt ein Versionsmanagement bereit. Alle Elemente erhalten eine Versionsnummer, die bei Änderungen automatisch erhöht wird. Der CMS-Redakteur kann bestimmen, welche Version in welcher Publikationsstufe veröffentlicht wird.

Neben der Unterscheidung von Versionen gibt es noch eine Unterscheidung nach verschiedenen Varianten. Die Varianten geben an, für welches Web ein Element (Content oder Navigationspunkt) gilt. Ein Element kann in mehreren Varianten existieren. So lassen sich unterschiedliche Inhalte für Intranet, Internet und Extranet oder mehrsprachige Webseiten realisieren.

Für die Verwaltung der Daten ist das Administrations-Tool (Admin-Tool), eine webbasierte Java-Anwendung, zuständig.

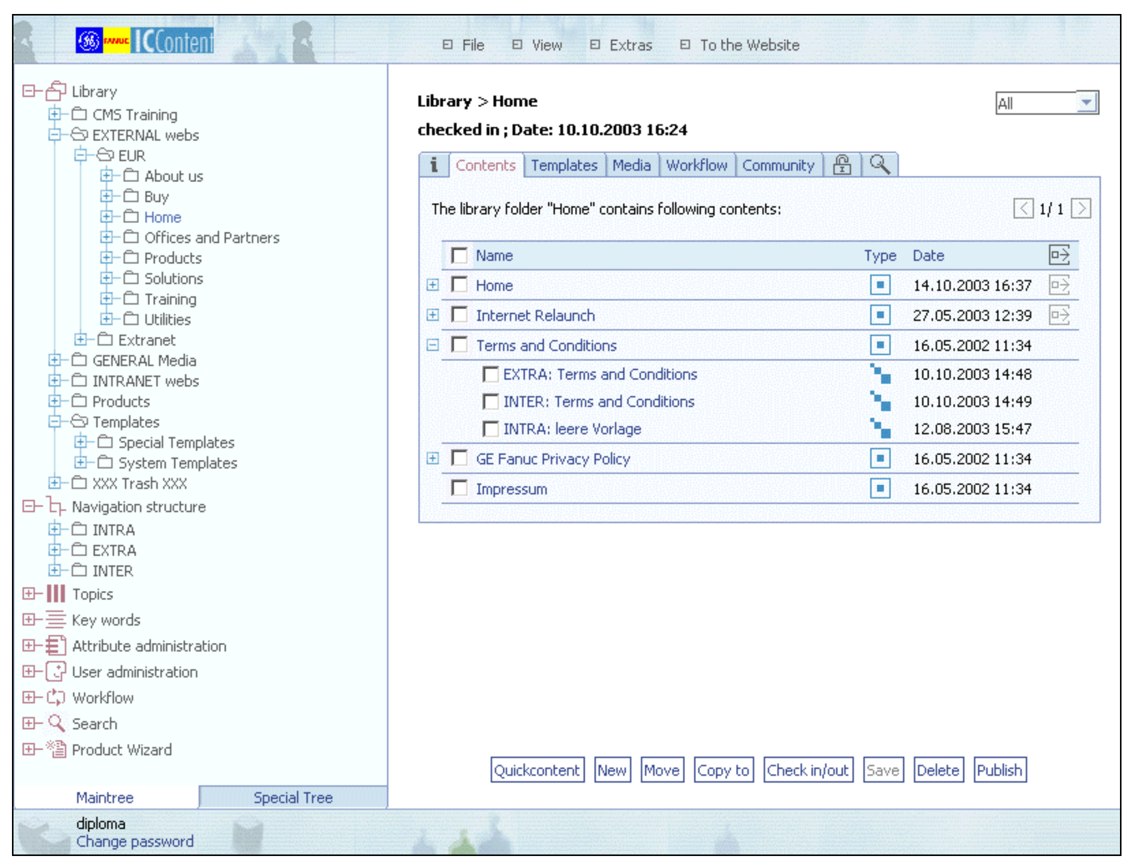


Abbildung 3-1: Admin-Tool von IC Content 4.0
Quelle: Screenshot des Admin-Tools

Nachdem Einloggen in das Admin-Tool erhält man einen Navigationsbaum, der die Daten in eine Struktur bringt. Mit Hilfe dieses Baumes navigiert man zum gesuchten Element (Content, Attribut, Template, Navigationspunkt, ...) und wählt anschließend die gewünschte Variante aus. Wenn man Zugriffsrechte auf das Element hat, werden alle relevanten Daten des Elements angezeigt. Um eine Änderung vorzunehmen, muss das Element ausgecheckt werden. Mit dem Auschecken wird eine neue Version

des Elements erzeugt. Um Änderungen an einem Element sichtbar zu machen, muss das Element eingecheckt und publiziert werden.
Zum Ansehen der Webausgabe dient die Java Server Page „broker.jsp“ auf dem jeweiligen Web. Mit Hilfe von verschiedenen Parametern werden Navigationspunkte, Templates und Contents angesteuert. Es wird reiner HTML-Code gesendet. Durch Anhängen der Zeichenkette „`&xml=true`“ an die eigentliche URL erreicht man, dass das XML-Dokument gesendet wird.
Eine typische URL einer CMS-Webseite sieht folgendermaßen aus: http://web.gefanuc.eur/cmsintra/broker.jsp?uMen=06f703d4-b0df-218cc0298603

Benutzerverwaltung und Rechtevergabe bei IC Content 4.0 basieren auf einem LDAP[16]-System.

Das Replikationssystem
Bei GE Fanuc werden zwei CMS-Systeme verwendet. Ein System liegt auf einem separaten Server mit jBoss und dient als Testsystem.
Das zweite CMS-System ist das Produktivsystem. Dieses System stellt die Daten für die Webs bereit. Zur Entlastung des Produktivservers läuft die Webausgabe für das Intranet auf einem zweiten Server, auf den die CMS-Datenbank des Produktivservers regelmäßig repliziert wird.
GE Fanuc betreibt eine Firewall, die Zugriffe in das interne Netzwerk von außen verhindert. Die Daten für das geplante CMS-basierte Inter- und Extranet müssen daher zusätzlich auf einen Server außerhalb des internen Netzwerkes repliziert werden.

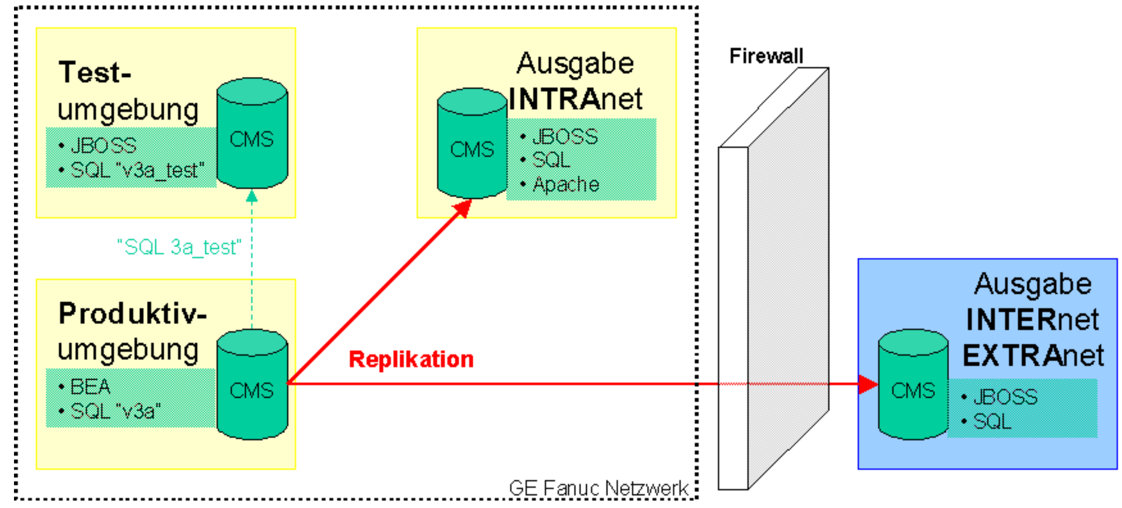


Abbildung 3-2: Server-Struktur des CMS-Systems
Quelle: GE Fanuc Automation

16 LDAP: Lightweight Directory Access Protocol
LDAP ist ein TCP/IP-basiertes Directory-Zugangsprotokoll, das sich im Internet und in Intranets als Standardlösung für den Zugriff auf Netzwerk-Verzeichnisdienste für Datenbanken, E-Mails, Speicherbereiche und andere Ressourcen etabliert hat.

3.2 Das CMS-basierte Intranet

Eine Intranetseite lässt sich in 5 Bereiche aufteilen: Kopf, linkes Menü, Untermenü, Contentbereich und Fußnote.

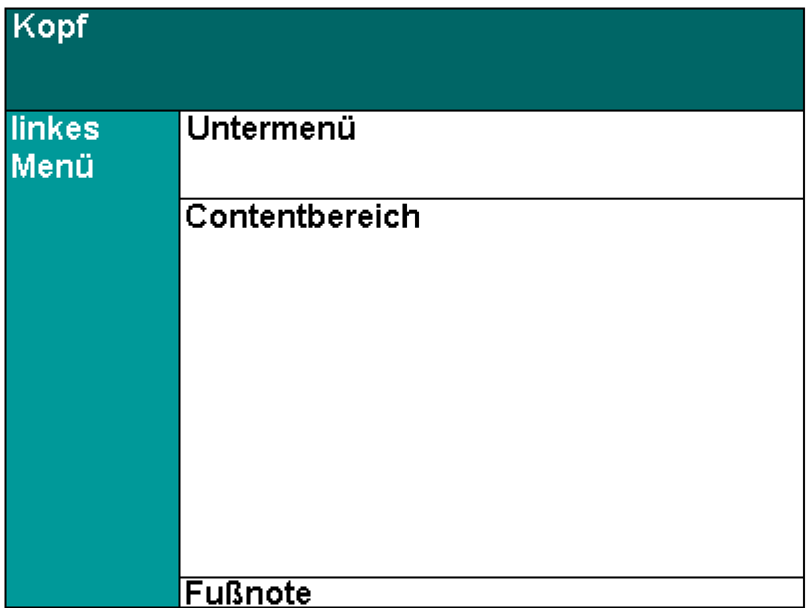


Abbildung 3-3: Bereiche einer Intranet-Seite
Quelle: eigene Darstellung

Der Kopf erstreckt sich am oberen Rand der Seite über die gesamte Breite und hat einen dunkelgrünen Hintergrund. Er enthält in der linken oberen Ecke das Unternehmenslogo, am rechten oberen Rand eine Roadmap, in der der Ast zum aktuellen Menüpunkt angezeigt wird. Der unteren Rand des Kopfes bildet das Abteilungsmenü.

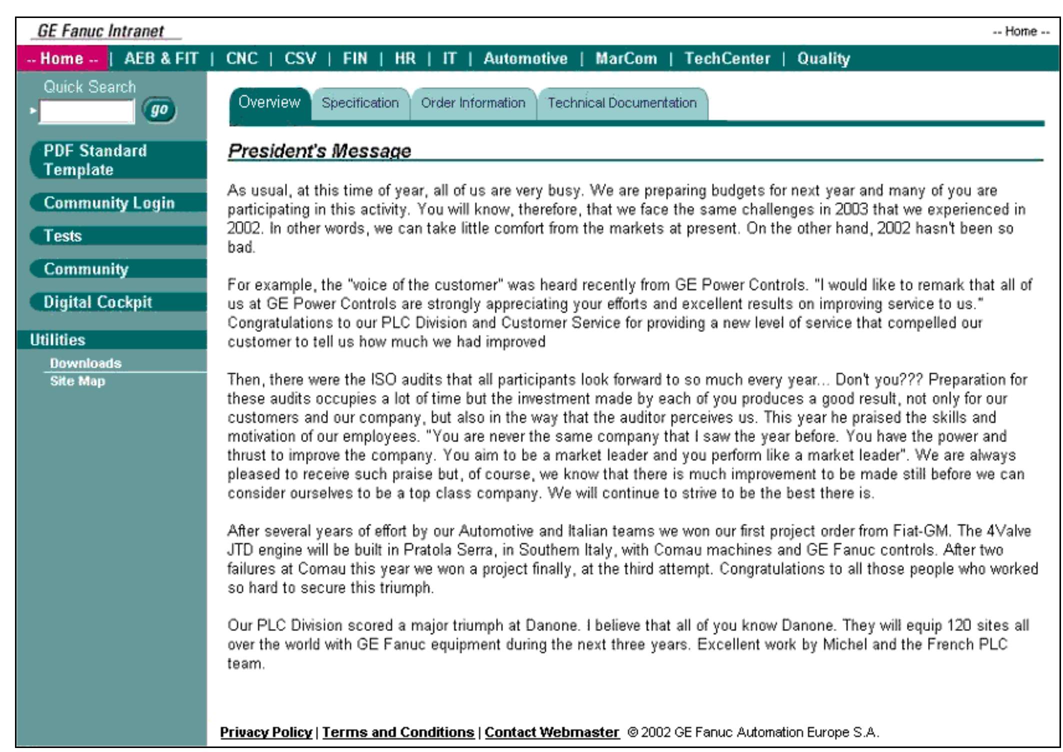


Abbildung 3-4: Altes Intranet-Layout
Quelle: GE Fanuc Intranet

Im linken Menü (hellgrüner Hintergrund) wird die oberste Menüebene der ausgewählten Abteilung angezeigt. Das Untermenü des letzten Menüpunktes wird immer angezeigt. Zusätzlich wird dort die Quicksearch-Funktionalität bereitgestellt. Diese bietet eine Volltextsuche über die Überschriften aller Intranet-Contents.

Das Untermenü beinhaltet Register- und Iconmenü. Es wird nicht bei jeder Seite angezeigt. Das Registermenü bildet die Menüpunkte als Register ab. Im Iconmenü wird das Untermenü des ausgewählten Registers angezeigt.

Im Contentbereich werden die Inhalte angezeigt, die mit dem aktuellen Menüpunkt verbunden sind.

In der Fußnote befinden sich spezielle Links zu Datenschutzbestimmungen, Geschäftsbedingungen und Kontaktformularen des Unternehmens und die Copyright-Information.

3.3 Analyse der verwendeten Templates

Teilziel dieser Diplomarbeit ist das Modularisieren der Templates, um deren Anzahl zu verringern und die Verwaltung zu erleichtern. Um dieses Ziel zu erreichen, müssen die vorhandenen Templates nach Einsatz und Aufbau untersucht werden.
Aus dieser Analyse werden Gemeinsamkeiten, Besonderheiten und Unterschiede hervorgehen, die die Basis zum Modularisierungskonzept bilden.

Von den existierenden Templates besteht keinerlei Dokumentation. Der erste Schritt besteht daher in der „Suche“ und Auflistung aller Templates.
Die Auflistung (siehe Anhang, S. 51) enthält Templatename, Position im CMS-Archiv und eine Beschreibung des Verwendungszwecks. Im Archiv (Library) des CMS existiert ein Ordner „Templates“, in dem alle Templates, möglicherweise auch in Unterordnern, liegen sollen. Diese Templates werden nacheinander gelistet.
Einige Templates liegen auch in anderen Teilen des Archivs. Dies wird erst festgestellt, als nach der Modularisierung einige Intranet-Seiten nicht mehr dargestellt werden.

Da mit der Modularisierung eine Veränderung oder Löschung der Templates bevorsteht, wird in diesem Schritt von jedem gelisteten Template eine Sicherheitskopie im lokalen Dateisystem angelegt.

Bei der Entscheidung welche Templates bei der Modularisierung berücksichtigt werden müssen, muss bekannt sein, welchen Zweck ein Template hat und „wo“ es verwendet wird.
In der Navigationsstruktur des CMS muss jedem Navigationspunkt ein Template zugewiesen („Anhängen eines Templates“) werden. Mit einem Durchlaufen der Navigationsstruktur im CMS wäre herauszufinden, ob und an welcher Position ein Template eingesetzt wird. Ein paralleles Durchlaufen der Webausgabe zeigt gleichzeitig die Funktion des verwendeten Templates.
Auf Grund der Menge der Navigationspunkte im Intranet (ca. 2200) ist diese Methode langwierig.

Eine Alternative dazu bieten Gespräche mit den CMS-Administratoren und falls nötig den CMS-Redakteuren einzelner Abteilungen. Die CMS-Administratoren sind für die Verwaltung und Pflege der Templates verantwortlich und haben einige selbst erstellt. Sie kennen die Funktionen der verschiedenen Templates. Die CMS-Redakteure legen die Templates für ihre Navigationspunkte fest: Sie wissen, welche Templates sie für welchen Zweck verwenden.

3.3.1 Einsatz der Templates beim allgemeinen Seitenaufbau

Da der Aufbau einer Intranetseite vom verwendeten Template abhängt erscheint es sinnvoll hier den Aufbau einer Intranetseite zu untersuchen.

Prinzipieller Aufbau
Das äußere Gerüst einer Intranetseite bilden drei Layer, die untereinander angeordnet sind. Im oberen Layer wird das GE Fanuc-Logo und die Roadmap dargestellt. Der zweite Layer enthält das Abteilungsmenü. Der letzte Layer beinhaltet das linke Menü, das Untermenü, den Contentbereich und die Fußnote. Innerhalb der Layer basiert das Layout auf einer Verschachtelung von Tabellen. Für die Darstellung der Navigation werden mehrere Grafiken eingebunden.

Darstellung der Menüs
Die Darstellung des linken Menüs und des Untermenüs erfolgt immer abhängig von der Position des aktuellen Navigationspunktes und dem daran angehangenen Template. Angezeigt wird immer der Name des Menüpunktes (vcDenotation). Für die Darstellung der Menüs gibt es vier Template-Typen:

- **Standard:**
  Der aktuelle Navigationspunkt wird im linken Menü angezeigt. Ist der Navigationspunkt direkter Kindknoten des Abteilungsmenüs, wird er als Oberpunkt dargestellt, sonst als Unterpunkt unter den Elternmenüs. Die Kindknoten werden als Unterpunkte des aktuellen Knotens dargestellt. Das Untermenü zeigt keine Knoten an.

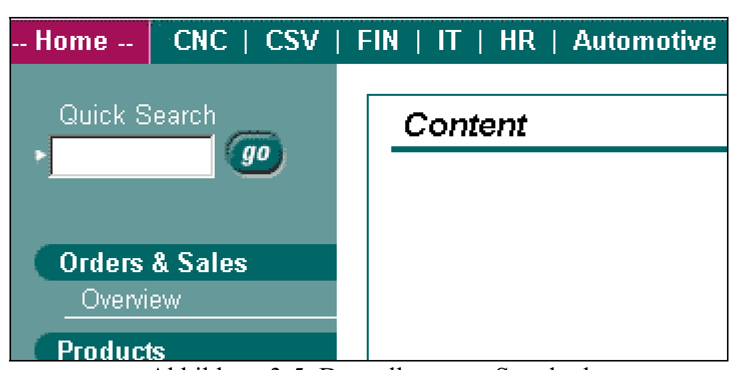


Abbildung 3-5: Darstellungstyp Standard
Quelle: bearbeiteter Screenshot einer Intranetseite

- **Switch:**
  Der aktuelle Navigationspunkt wird im linken Menü angezeigt. Ist der Navigationspunkt direkter Kindknoten des Abteilungsmenüs, wird er als Oberpunkt dargestellt, sonst als Unterpunkt unter den Elternmenüs. Die Kindknoten werden im Registermenü des Untermenüs dargestellt. Das Iconmenu zeigt keine Navigationspunkte an.

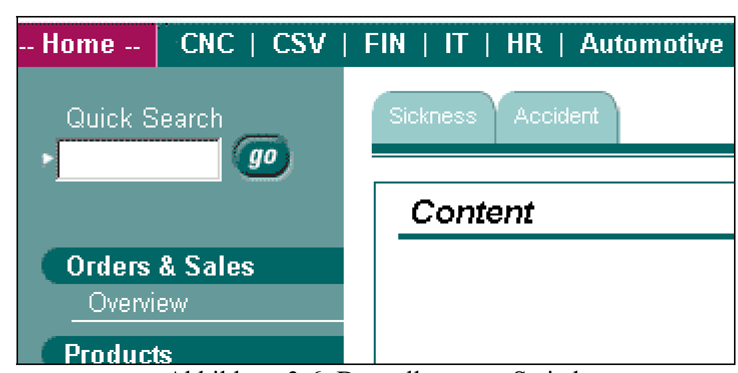


Abbildung 3-6: Darstellungstyp Switch
Quelle: bearbeiteter Screenshot einer Intranetseite

- **Register:**
  Der aktuelle Menüpunkt ist Teil des Registermenüs. Alle Elternmenüs bis zum Abteilungsmenü werden im linken Menü dargestellt. Die Kindknoten werden im Iconmenü dargestellt.

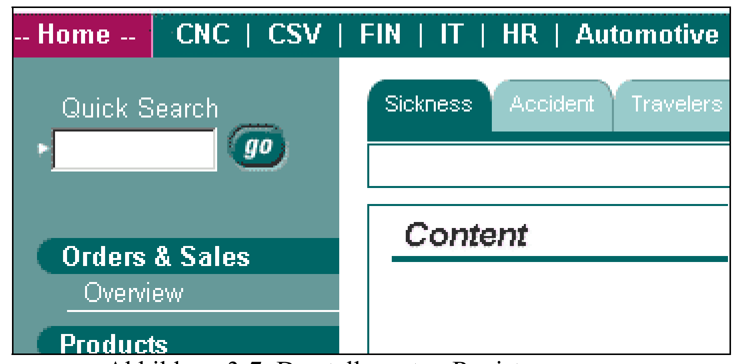


Abbildung 3-7: Darstellungstyp Register
Quelle: bearbeiteter Screenshot einer Intranetseite

- **Icon:**
  Der aktuelle Menüpunkt ist Teil des Iconmenüs. Das direkte Elternmenü wird im Registermenü angezeigt, die weiteren in der linken Navigation. Kindknoten können nicht angezeigt werden. Jedem Menüpunkt ist ein Icon zugeordnet (daher der Name Iconmenü).Die Menüpunkte sind „fließend" von links nach rechts angeordnet. Je nach Anzahl der Menüpunkte werden mehrere Zeilen dargestellt.

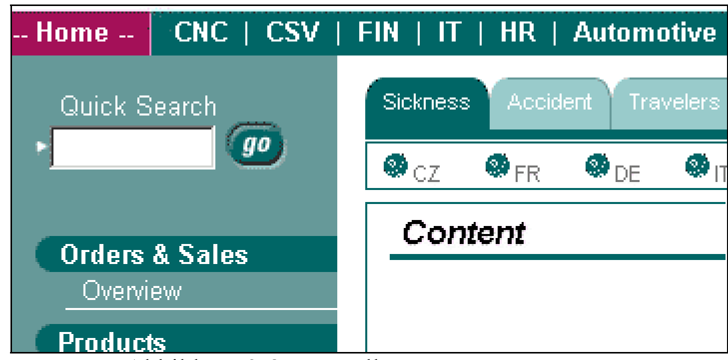


Abbildung 3-8: Darstellungstyp Icon
Quelle: bearbeiteter Screenshot einer Intranetseite

Die Templates „Quality Register ISO 9000" und „Quality Icon ISO 9000" stellen die Menüpunkte des Iconmenüs untereinander, anstatt nach rechts fließend, dar.

Für diese Typen stehen jeweils mehrere Templates zur Auswahl. Diese unterscheiden sich in der Darstellung des Contents.

Darstellung des Contents
Für jede der vier Menü-Varianten steht ein Template ohne Erweiterung zur Verfügung („Standard", „Switch", „Register" und „Icon"). Die Inhalte, die im CMS vorliegen müssen, werden untergliedert in Überschrift (Meta-Information „vcHeadline") und Contenttext, der im HTML-Format im CMS liegt, dargestellt. Die Templates können beliebig viele Contents auf einer Seite anzeigen.

Die Templates mit der Erweiterung „without header" zeigen nur den Contenttext an. Sie sind für Contents ohne Überschrift gedacht.

Templates mit der Erweiterung „ExLink" bauen ein Frameset mit 3 oder 4 Frames auf. Jeweils ein Frame für Kopf, linke Navigation, falls benötigt für das Untermenü und für den Content. In den Content-Frame wird eine Webseite geladen, die nicht im CMS (externer Content) verwaltet wird. Diese wird über die Attribut-Klasse „ExLink" in einem Content festgelegt.

Die Templates mit der Erweiterung „ExLink PopUp" stellen externen Content in einem PopUp-Fenster dar. Der Contentbereich der CMS-Seite bleibt leer.

Das Template News baut eine Standard-Navigation auf. Von den Contents wird nur die Überschrift und die Kurzbeschreibung (Meta-Information „txtShortText") in einer Box angezeigt. Falls für einen Content angegeben werden auch die Informationen der Attribut-Klasse „Author" angezeigt.

Das Template „Register Table Attributes" erzeugt eine Tabelle der mit einem Content verbundenen Attribute. Über diese Darstellung stellen einige Abteilungen eine Projektübersicht im Intranet bereit.

In der Intranet-Variante des CMS existieren 73 Templates, von denen 46 genutzt werden.

3.3.2 Struktur der Templates

Nach dem alle Templates gelistet sind und bekannt ist, welche Templates bei der Modularisierung zu berücksichtigen sind, muss die Struktur dieser Templates untersucht werden, um Unterschiede und Gemeinsamkeiten herauszufinden.

Kategorisierung der Templates
Die Templates können nach verschiedenen Kriterien kategorisiert werden. Für einen CMS-Redakteur ist es wichtig zu wissen, welche Templates er mit Menüpunkten verknüpfen kann. Templates, die dieses Kriterium erfüllen, werden Start-Templates genannt. Neben den Start-Templates existieren Frame-Templates und Spezial-Templates, die bei einer Verknüpfung zu Transformationsfehlern führen. Die Start-Templates können weiter unterteilt werden.

- **Start-Templates**
  Die „normalen“ Start-Templates enthalten eine Reihe von Template-Regeln, für den Aufbau des Kopfes, der linken Navigation, des Untermenüs und des Contents. Die meisten dieser Regeln sind in allen Templates identisch. Bei einigen Regeln gibt es kleine Abweichungen, die meist nur wenige Zeilen betreffen. Größere Unterschiede treten in den Template-Regeln auf, die für die Darstellung des Contents verantwortlich sind. Es gibt nur eine Template-Regel, die über das „match“-Attribut (siehe Kapitel 2.1.3) angesprochen werden. Alle weiteren Regeln werden über eine Call-Template-Anweisung aufgerufen.
  Die Start-Templates enthalten eine Importanweisung, die das System-Template „import“ einfügt. Es gibt eine externes CSS. Dies wird allerdings nur teilweise verwendet, da auch viele Formatierungs-Anweisungen „hart“ eincodiert sind.
  Jedes Start-Template hat eine Länge von 600 bis 750 Zeilen.
- **Start-Frame-Templates**
  Um externen Content im Contentbereich darzustellen muss die Seite in ein Frameset unterteilt werden. Die Templates dieser Kategorie enthalten nur eine Template-Regel. In ihr wird das Frameset aufgebaut. Die URL der Seite, die im Contentbereich angezeigt werden soll wird beim „Standard Frame“-Template aus einer Meta-Information des Navigationspunktes () ausgelesen, bei den anderen Templates aus dem Attribut „exlink“, welches an einen (normalerweise leeren) Content angehangen ist. Für die weiteren Frames werden je nach Template-Typ (Standard, Switch, Register, Icon) verschiedene Seiten aus dem CMS geladen. Die URL ist die einer gewöhnlichen CMS-Seite. Als Parameter wird die ID des Templates und die ID des gewählten Menüpunktes angehangen.
- **Start-Spezial-Templates**
  Die Templates dieser Kategorie haben verschiedene Funktionen und einen teilweise stark unterschiedlichen Aufbau. Das Template „SiteMap“ erstellt mit den üblichen Regeln die Navigationsstruktur, verwendet aber spezielle Regeln zur Darstellung des Contentbereichs. Diese Templates sind für spezielle Anwendungen programmiert, die oftmals nur auf einer einzigen Seite des Intranets benötigt werden.
- **Frame-Templates**
  Zur Gruppe der Frame-Templates gehören die Templates, die von einem Start-Frame-Template aufgerufen werden. Sie enthalten Regeln, um den Teil der Seite, der durch den entsprechenden Frame dargestellt wird zu erzeugen. Für das linke Menü und das Untermenü werden je nach Template-Typ unterschiedliche Templates verwendet, die sich in ihren Regeln leicht unterscheiden.
- **Spezial-Templates**
  Diese Templates erfüllen wie die Start-Sonder-Templates verschiedene Funktionen und haben daher einen unterschiedlichen Aufbau. Sie werden von anderen Templates importiert.

Es fällt auf, dass die Template-Regeln zum Aufbau des Kopfbereiches in allen wichtigen Templates identisch sind. Die Regeln zum Aufbau des linken Menüs unterscheiden sich je nach Typ in ein paar Zeilen. Die Regeln zum Aufbau des Untermenüs werden beim Typ „Standard“ nicht benötigt. Differenzen bei den anderen Typen sind begrenzt auf wenige Zeilen. Die Darstellung der Fußnote wird

bei allen Templates, die eine Fußnote aufbauen durch eine importierte Regel aus dem Template „import“ erzeugt. Größere Differenzen enthalten die Template-Regeln zur Darstellung des Contentbereichs, sowie einige Start-Spezial-Templates.

Die Regeln für das Untermenü sind so programmiert, dass Grafiken des Menüs auch dann sichtbar sind, wenn es keine Menüpunkte gibt.

Das Code-Fragment zur Erzeugung des Links für einen Menüpunkt ist in allen Template-Regeln die einen Menüpunkt erzeugen identisch.

Die Differenzen beziehen sich im Großteil der Templates auf drei Punkte:

- Aufbau der Menüs (abhängig vom Typ des Templates)
- Darstellung des Contentbereichs
- Ausrichtung der Menüpunkte im Iconmenü

3.4 Resultierende Probleme

Überarbeitung des Layout
Das Intranet soll ein neues Design erhalten. Dabei sollen möglichst wenig Grafiken zur Menüdarstellung verwendet werden. Durch eine Umgestaltung des Kopfbereichs soll mehr Raum für den Content geschaffen werden.
Da viele der Layoutanweisungen in den Template-Regeln programmiert wurden und diese in vielen Templates vorhanden sind, müssten Änderungen an vielen Templates ausgeführt werden. Für eine Änderung im linken Menü müssten 27 Templates geändert werden, für eine Änderung des Registermenüs wären dies immer noch 20 Templates.

Einpflegen neuer Funktionen

- Im Intranet soll eine Community-Funktion integriert werden. Dadurch wird eine Unterteilung der CMS-Benutzer in verschiedene Gruppen möglich, die unterschiedliche Berechtigungen zum Ansehen verschiedener Contents besitzen. Dazu muss im linken Menü ein Bereich für die benötigte Login-/Logout-Funktion integriert werden.
- Um die Orientierung zu erleichtern, sollen die geöffneten Menüpunkte farblich hervorgehoben werden.
- Content die im „News“-Format dargestellt werden, sollen nach Erstellungsdatum sortiert und neue News zusätzlich hervorgehoben werden.
- Zusätzlich soll es möglich sein über eine Werkzeugleiste unterhalb der Contents die folgenden Funktionen zu starten.
    - PDF-Creation: Aus der aktuellen Seite wird ein PDF-Dokument erzeugt.
    - Print-View: Die aktuelle Seite wird in einem „druckerfreundlichen“ Layout dargestellt.
    - SendURL: Die aktuelle URL kann per Email verschickt werden.
    - Author-Information: In einem PopUp-Fenster werden Informationen über die Autoren der aktuellen Seite angezeigt.

Auch hier tritt die Problematik auf, dass sehr viele Templates zu ändern sind, um die neuen Funktionen zu integrieren.

Vermischung von Frameseiten und Seiten ohne Frames
Standardmäßig werden die Intranet-Seiten ohne Frames dargestellt. Zur Darstellung externer Contents wurden Framesets erzeugt. Es kann nur ein Content angezeigt werden, eine Vermischung von Contents ist nicht möglich. CMS-Redakteure müssen die Verlinkung auf einen externen Content, evtl. schon in anderen Contents oder in Navigationspunkten berücksichtigen, um korrekte Links einzupflegen und das richtige Template auszuwählen. Für die Darstellung externer Contents werden 17 Templates verwendet, was über 35% der gesamten verwendeten Templates ausmacht.

PopUp-Fenster
In einem PopUp-Fenster können nur externe Contents dargestellt werden. Durch die Programmierung der Template-Regel zum Öffnen eines PopUp-Fensters ist es nicht möglich Attribute mit Ausnahme des Exlink-Attributes zu verwenden.

Übersicht für die Redakteure
Ein CMS-Redakteur muss die Navigationspunkte mit einem Template verknüpfen. Dies wird im Admin-Tool des CMS über ein DropDown-Menü gesteuert. In diesem Menü werden dem Redakteur alle im CMS existierenden Templates nur mit Namen, teilweise abgeschnitten, angezeigt.
Es fällt einem Redakteur schwer das Template auszuwählen, das auf seinen Anwendungsfall passt. Eine Auswahl mit Hilfe einer detaillierten Liste kann unter Umständen aufwendig sein.

Übertragbarkeit auf Inter- und Extranet
Das Intranet stellt für GE Fanuc nur den ersten Schritt in Richtung CMS dar. Für 2003 ist die Integration der Website www.gefanuc-europe.com (Internet) als CMS-Variante und das Erstellen einer Extranet-Variante im CMS-Basis geplant. Für die neuen Varianten müssen Templates bereitgestellt werden. Da die verschiedenen Varianten vom Seitenaufbau sehr ähnlich sein sollen, bietet sich eine Generierung der benötigten Templates aus den Intranet Templates an. Dadurch würden die beschriebenen Probleme auf die neuen Webs übertragen werden.
Das Erarbeiten eines neuen Template-Konzeptes für Extra- und Internet, bei Beibehaltung der Intranet-Templates, erscheint ebenfalls nicht sinnvoll, da dies eine zusätzliche „Verwirrung“ der CMS-Redakteure bedeuten würde.

4 Neukonzeption und Überarbeitung der Templates

Wichtigstes Ziel der Neukonzeptionierung ist die Modularisierung der Templates. Dadurch wird die Templateanzahl reduziert, was die Arbeit der CMS-Redakteure und die Verwaltung der Templates vereinfacht. Dabei müssen die bisherigen Funktionen, sowie die neuen Funktionen und die Veränderung des Layouts berücksichtigt werden. Templates die im aktuellen Intranet nicht verwendet werden bleiben unberücksichtigt.

4.1 Konzept-Erarbeitung

Die Überarbeitung der Templates wird in 5 Arbeitsschritten durchgeführt werden:

1. Modularisierung
2. Überarbeitung der Seitenstruktur und des Designs
3. Integration neuer Funktionen
4. Dokumentation für die Redakteure, Anwender und Administratoren
5. Relaunch der neuen Templates

Die Neukonzeption wird in Template-Konzept und Layout-Konzept unterteilt.

4.1.1 Template-Konzept

- **Zentraler Menüaufbau**
  Die Template-Regeln zum Aufbau der Menüs sind in allen Templates eines Template-Typs identisch. Zwischen den Regeln der verschiedenen Template-Typen gibt es nur minimale Änderungen. Eine zentrale Ablage dieser Template-Regeln ist sinnvoll. Fast identische Template-Regeln werden zu einer einzigen mit entsprechenden Verzweigungsanweisungen für die abweichenden Code-Fragmente.

- **häufige Content-Darstellungen zentralisieren**
  Templates für bestimmte Content-Darstellungen gibt es meist für mehrere Template-Typen oder sind in Zukunft für mehrere Template-Typen denkbar. Dazu gehören zum Beispiel die normale Darstellung und die News-Darstellung. Die dafür benötigten Template-Regeln werden im zentralen Template abgelegt

- **Parameter-System**
  Zur Steuerung des Programmablaufs im zentralen Template werden Parameter benötigt. Für den Menüaufbau werden 2 Parameter (Type, IconNav), für die Auswahl der Content-Darstellung ein Parameter (Content) vorgesehen.
  Der Parameter Type enthält einen Wert, der den Template-Typ (Standard, Switch, Register oder Icon) repräsentiert (Kapitel 3.3.1, S.20), IconNav legt die Anordnung der Menüpunkte im Iconmenü fest. Der Parameter Content repräsentiert die verschiedenen Content-Darstellungen.
  Um die Parameter im zentralen Template auszulesen gibt es zwei sinnvolle Ansätze:
  1. Die Parameterwerte werden in einer Meta-Information des Navigationspunktes als Zeichenkombination hinterlegt und vom zentralen Template ausgelesen. Da Navigationspunkte keine selbstdefinierten Attribute erlauben, muss eine bereits existierende Meta-Information zweckentfremdet

werden. Für das Darstellen eines Content-Links[17], die im Normalfall als Typ Standard angezeigt werden, müsste ein zusätzliches Template erstellt werden, das das zentrale Template importiert und die Parameterwerte übergibt. Der große Vorteil dieses Ansatzes liegt darin, das für die wichtigsten Darstellungen nur ein Template benötigt wird.

2. Es werden für jeden der gebräuchlichsten Anwendungsfälle Start-Templates erstellt. Diese enthalten die Parameter-Werte. Das zentrale Template wird in die Start-Templates importiert. Die Template-Regeln des zentralen Templates lesen zur Programmfluss-Kontrolle die Parameter-Werte aus. Die Anzahl der benötigten Templates ist bei dieser Methode höher. Der Vorteil besteht darin, dass keine Meta-Informationen „belegt“ werden, die später evtl. für andere Zwecke gebraucht werden.

Für Navigationspunkte im CMS gibt es nur ein unbenutztes Feld für Meta-Informationen, das für eine Kommentierung des Navigationspunktes vorgesehen ist. Um bei Content-Links flexibler zu sein, müssen zusätzliche Templates erstellt werden, die den Start-Templates des zweiten Ansatzes entsprechen. Bei gleicher Flexibilität spart der erste Ansatz lediglich ein Template, schränkt jedoch die weitere Benutzung der Kommentar-Information der Navigationspunkte ein. Daher wird der zweite Ansatz verfolgt. Da das zentrale Template von den Start-Templates importiert wird es als **Import-Template** bezeichnet.

- **Template-Kategorien**
  Die überarbeiteten Templates werden in **Start-** und **System-Templates** unterteilt. Die Start-Templates können in **Standard-** und **Spezial-Templates** unterteilt werden.
  Die im vorherigen Abschnitt beschriebenen Start-Templates gehören zur Kategorie der Standard-Templates. Neben den Parameterwerten enthält ein Standard-Template die Importanweisung für das Import-Template, und eine Template-Regel (Start-Regel), deren einzige Anweisung der Aufruf einer Template-Regel (Master-Regel) aus dem Import-Template ist. Aus der Master-Regel werden alle weiteren Template-Regeln aufgerufen.
  Bei der Analyse der Templates in Kapitel 3.3.2 (S. 22) wurden die Start-Spezial-Templates erwähnt. Die Template-Regeln zum Aufbau der Menüs entsprechen denen der „alten“ Start-Templates. Die speziellen Regeln für die Content-Darstellung sind einmalig. Diese Templates gehören zur Kategorie der Spezial-Templates. Spezial-Templates haben fast den gleichen Aufbau wie Standard-Templates. Template-Regeln, die sich im Aufbau von denen des Import-Templates unterscheiden, werden im Spezial-Template, mit gleichem Regel-Namen, programmiert. Aufgrund das Konflikt-Konzepts (siehe Kapitel 2.1.3, S. 10) der XSLT-Prozessoren bei gleichnamigen Template-Regeln werden die Regeln des Import-Templates „überschrieben“.
  Standard- und Spezial-Templates dürfen von den CMS-Redakteuren mit Navigationspunkten verknüpft werden.
  Das Import-Template darf nicht von CMS-Redakteuren mit einem Navigationspunkt verknüpft werden, sondern wird vom CMS selbst aufgerufen. Aus diesem Grund zählt das Import-Template zur Kategorie der System-Templates.

[17] es wird kein Menüpunkt aufgerufen, sondern ein Content kombiniert mit einem Template

- **keine Framesets**
  Als Alternative zu einem Frameset zur Darstellung externer Contents bietet HTML mehrere Möglichkeiten an.
  Die flexibelste Lösung stellt das object-Element dar, mit dem sich externe Objekte (Dateien) in eine HTML-Seite einbinden lassen. Zudem gibt es die Möglichkeit, bei einem nicht darstellbaren Objekt alternative Objekte oder einen alternativen Text anzuzeigen. Obwohl das object-Element vom Microsoft Internet-Explorer ab Version 3 und von Netscape ab Version 4 unterstützt wird, treten bei den neuesten Browsern (IE 5.5, Netscape 7) teilweise Probleme auf[18].
  Eine für diesen Fall bessere Lösung stellt das iFrame-Element dar. In der HTML-Seite wird ein „inline frame“[19] erstellt, in den eine externe Datei geladen werden kann. Das iFrame-Element wird ab Version 3 des Internet-Explorers und Version 6 von Netscape unterstützt[20]. Die im Intranet genutzten externen Contents sind mit dieser Alternative problemlos darstellbar.
  Templates, die Framesets erzeugen, oder zum Laden in Frames gedacht sind werden nicht mehr benötigt.

- **Flexiblere Content-Darstellung**
  Durch eine Erweiterung der Attribut-Klasse „EXlink“ und ein gezieltes Auslesen der Attribute wird es möglich sein, CMS-basierte und externe Contents in einem PopUp-Fenster oder einem eingebetteten Rahmen darzustellen. CMS-basierte Contents können zudem als „echtes“ Element der Intranet-Seite dargestellt werden. Eine beliebige Kombination der Contents und den angesprochenen Darstellungsformen ist möglich.
  Neben dem bestehenden Attribut „exlink“, das die URL eines externen Contents enthält, wird das Attribut „popup“ definiert, über das festgelegt werden kann, ob die URL in einem Rahmen auf der Seite, oder in einem PopUp-Fenster angezeigt wird. Um auch interne Contents auf diese Weise darzustellen, kann für das „exlink“ Attribut der Wert „this“ eingetragen werden.

- **Zusätzliche Parameter im Import-Template**
  Werte des XML-Streams, die häufiger verwendet werden, werden im Import-Template als Parameter definiert. Der Parameter-Wert wird durch XSL-Anweisungen dynamisch erzeugt. Zu diesen Parametern gehört z.B. die URL des aktuellen Menüpunktes und der Seitentitel. Weitere Parameter, die dynamisch erzeugt werden dienen dem Aufbau der Navigationsstruktur.

- **nachvollziehbare Namensgebung**
  Die CMS-Redakteure müssen ihre Menüpunkte mit einem Template verknüpfen. Um die Auswahl des richtigen Templates möglichst einfach zu gestalten, müssen die Templates eindeutige Namen haben, aus denen die Funktion des Templates erkennbar ist.

[18] Vgl. Münz, SelfHTML; URL: http://selfhtml.teamone.de/html/multimedia/objekte.htm (13.11.2003)
[19] inline Frame: Rahmen im Textfluss, eingebetteter Rahmen
[20] Vgl. Münz, SelfHTML; URL: http://selfhtml.teamone.de/html/frames/eingebettete.htm (13.11.2003)

4.1.2 Layout-Konzept

Eine Überarbeitung des Layouts ist ebenfalls Bestandteil der Template-Überarbeitung.

- **Tabellen-basiertes Layout**
  Um die Struktur der Seite für die Administratoren möglichst übersichtlich zu gestalten, wird ein tabellen-basiertes Layout verwendet, das die Struktur bis ins Detail festlegt. Ein Ausnahme bildet das nach rechts fließende Iconmenü, das über div-Elemente erstellt wird, um eine flexible Anordnung zu ermöglichen. Die 5 Bereiche (siehe Kapitel 3.2) einer Intranetseite bleiben bestehen, wobei sich Untermenü, Contentbereich und Fußnote zum Hauptbereich (siehe Abbildung 3-3 , S. 18) zusammenfassen lassen.

- **Trennung von Struktur und Design**
  Um eine spätere Änderung des Designs zu erleichtern und die Templates leicht auf Inter- und Extranet übertragbar zu machen werden keinerlei Design-Anweisungen in den Templates verwendet. Das Design wird in einem CSS definiert, das in den transformierten HTML-Code eingebunden wird. Dazu müssen die Elemente im HTML mit einem Class- oder einem ID-Attribut versehen sein. Um das Design möglichst flexibel zu machen, wird jede Tabelle und jede Tabellenzelle mit einem eindeutigen Class-Attribut versehen. Um die Hervorhebung der geöffneten Navigationspunkte zu realisieren, werden die Zellen von Menüpunkten zusätzlich mit einer ID versehen, die dynamisch gesetzt wird.

- **Umfangreiches, erweiterbares CSS**
  Die CSS-Datei wird nicht in das CMS integriert, sondern in einem Verzeichnis auf dem Server abgelegt. Dies hat den Vorteil, dass auch andere Anwendungen (externe JSP's, externe Contents, ...) die CSS verwenden können.
  Neben den Style-Anweisungen für das Seitengerüst, enthält die CSS auch eine Reihe von Anweisungen, die von den CMS-Redakteuren in Contents verwendet werden können.

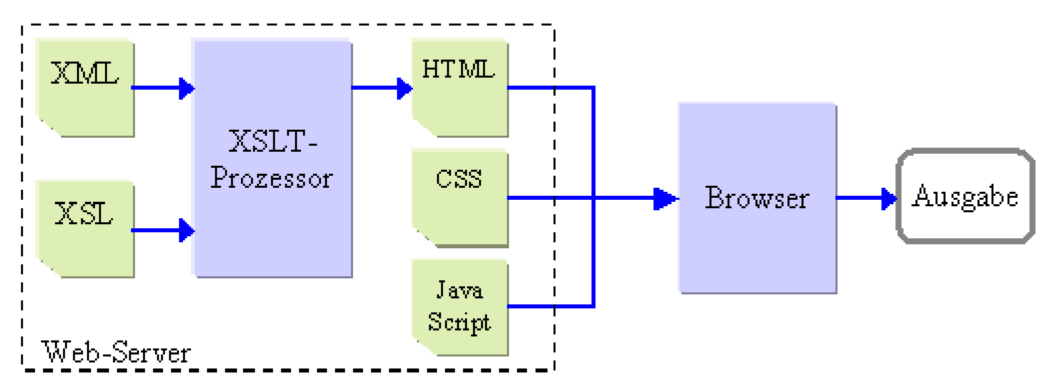


Abbildung 4-1: Entstehungsprozess der Webausgabe
Quelle: eigene Darstellung

Zusammenfassung
Aus den beschriebenen Konzepten ergeben sich die folgenden Kernpunkte der Template-Überarbeitung:

- 3 Template-Kategorien: Standard, Spezial und System
- Zentralisierte Template-Regeln für alle Start-Templates
- kurze, einfache Start-Templates (Standard und Spezial)
- Komplett tabellen-basiertes Layout
- Trennung zwischen Struktur und Design
- Umfangreiches erweiterbares CSS
- Einbindung neuer Funktionalitäten
- Einbindung externer Contents ohne Erstellung von Framesets
- Darstellung von externen und CMS-basierten Contents auf einer Seite
- Optionale PopUp-Fenster für externen und CMS-basierten Content neben der CMS-Seite
- Reduzierung der Templateanzahl
- Einfacheres Erstellen neuer Templates

Das Resultat der Modularisierung ist eine deutliche Reduzierung der benötigten Templates, was zu einer einfacheren Handhabung und einem geringeren Verwaltungsaufwand führt.

4.1.3 Implementierung neuer Funktionen

Neben der Überarbeitung der bestehenden Funktionen werden auch neue Funktionen implementiert, die hier kurz beschrieben werden. Einige der neuen Funktionen werden durch Anklicken eines Icons aufgerufen. Diese Icons werden in der Toolleiste zusammengefasst, die zwischen dem letzten Content und der Fußnote angeordnet wird.

- **Hervorhebung der geöffneten Menüpunkte**
  Um die Orientierung im Intranet zu erleichtern, werden die geöffneten Menüpunkte aller Menüs farblich hervorgehoben.

- **Hervorhebung aktueller News und automatische Sortierung**
  Um Neuigkeiten in der Content-Darstellung „News“ besser erkennbar zu machen, werden aktuelle Beiträge für 10 Tage speziell hervorgehoben. Dazu ist es nötig, dass die Autoren-Informationen (Attribut-Klasse „Author-Information“) ausgefüllt sind.
  Die Reihenfolge der News-Beiträge war bisher abhängig von deren Reihenfolge im CMS. Neue Beiträge mussten „per Hand“ an die erste Position gestellt werden. Eine XSL-Sortierfunktion wird die Beiträge zukünftig nach Erstellungsdatum sortieren.

- **Druckvorschau**
  Um beim Ausdrucken von Webseiten möglichst viel Raum für die Contents zu schaffen kann mit Hilfe eines System-Templates eine „druckerfreundliche“ Seite, die in einem neuen Fenster geöffnet wird, erzeugt werden. Sie enthält keinerlei Navigationselemente. Eine Angabe der URL und der Roadmap der aktuellen Seite am unteren Ende der Druckvorschau zeigen die Quelle des Dokumentes an. Die Funktion kann durch ein Icon in der Toolleiste ausgeführt werden.

Die Realisierung dieser Funktion wirft ein Problem auf: das Layout externer Contents kann nicht beeinflusst werden.
Nach Abwägen von Aufwand und Nutzen wurden Abstriche in der Nutzbarkeit dieser Funktion gemacht. Auf einer Seite mit rein CMS-basierten oder gemischten Contents wird das Template „PrintView“ aufgerufen. Es wird ein spezielles Layout erstellt, bei dem ein Minimum an Farben verwendet wird. Zusätzlich wird ein Banner am Beginn der Seite, und die URL zur Intranetseite, sowie das Erstellungsdatum am Ende der Seite eingefügt. Es werden nur CMS-basierte Contents, mit Autor-Informationen, angezeigt. Bei Seiten die einen (nur einen) externen Content, keine CMS-basierten, enthalten wird der Content in einem neuen Fenster, ohne umgebende Menüs, geöffnet. Für Seiten die mehr als einen externen Content und keinen CMS-basierten Content enthalten steht diese Funktion nicht zur Verfügung.

- **Umwandlung einer Intranet-Seite in ein PDF-Dokument (PDF-Funktion)**
  Für diese Funktion, die über ein Icon der Toolleiste ausgeführt wird, hat die Firma IC Team eine JSP[21]-Applikation (PDF-Tool) entwickelt. Es wandelt eine HTML-Seite in ein PDF-Dokument um. Die URL des HTML-Dokuments wird als Parameter an die JSP übergeben. Da auch bei PDF-Dokumenten möglichst viel Raum für den Content vorhanden sein soll, wird das PDF-Tool mit der URL der Druckvorschau aufgerufen. Bei der Verarbeitung externer Contents kann es zu Fehlern des PDF-Tools kommen. Daher wird die PDF-Funktion für Seiten, die nur externe Contents enthalten nicht zur Verfügung stehen. Bei Seiten, die gemischte Contents enthalten, werden nur CMS-basierte Contents ins PDF übernommen.

- **Anzeige von Autoren-Informationen**
  Die dritte Funktion in der Toolleiste zeigt die Autor-Informationen für die Contents einer Intranet-Seite an. In einem PopUp-Fenster werden die Attribut-Werte der Attribut-Klasse „Author-Information“ angezeigt. Eine zusätzliche Anzeige der jeweiligen Überschrift erleichtert die Zuordnung. Für externe Contents stehen keine Autoren-Informationen zur Verfügung.

- **Versand einer Seiten-URL per Email (Send2Friend)**
  Über die Send2Friend Funktion wird ein Fenster zum Verfassen einer E-Mail mit dem Standard-E-Mail-Programm (z.B. MS Outlook) geöffnet. Ein Text, der die URL und den Titel der Intranetseite enthält, und ein Betreff sind bereits vorgefertigt. Es ist noch die Email-Adresse des Empfängers einzutragen.
  Die ursprüngliche Idee, diese Funktion über ein Formular zu realisieren wurde verworfen, da die Mail von einem GE Fanuc Server erzeugt worden wäre und somit der Auslöser (Absender) der Funktion nicht direkt erkennbar wäre.

- **Sortierung von Attribut-Tabellen und Einbindung des „Projectwizard“**
  Attribut-Tabellen, die bisher durch Templates erzeugt wurden hatten eine willkürliche Reihenfolge der Attribute und keine Sortierung. Ein zusätzlicher Parameter in den Start-Templates (AttrSort) dient dazu, die Reihenfolge der Attribute festzulegen. Über den Parameter lässt sich bestimmen, welche Attribute angezeigt werden und in welcher Reihenfolge. Die Sortierung der Contents geschieht nach dem Wert des ersten angezeigten Attributs.

[21] JSP: Java Server Pages Online im Internet; URL: http://java.sun.com/products/jsp/ (13.11.2003)

Der Hauptanwendungsfall für Attributtabellen sind Projekt-Tabellen, in denen die Abteilungen eine Übersicht ihrer Projekte bieten. Zur Verwaltung dieser Tabellen, hat IC Team eine Applikation, den Projectwizard, bereitgestellt. Der Projectwizard ist eine browserbasierte Anwendung. Um den Projectwizard aus der Projekt-Tabelle aufrufen zu können, wird ein weiteres Attribut (ProjectToolLink) in die Start-Templates eingefügt, das die URL des Projectwizards für die jeweilige Anwendung enthält.
Da die Templates für die Attribut-Tabellen spezifisch für einen Anwendungsfall erstellt werden, gehören sie zur Kategorie der Spezial-Templates.
Die Funktion zur Sortierung und der Link zum Projectwizard müssen in die Template-Regeln für die Darstellung von Attribut-Tabellen integriert werden.

- **PopUp-Menüs für die erste Ebene unter dem Abteilungsmenü**
  Um die Navigation zu erleichtern wird eine PopUp-Menü-Funktion in das Abteilungsmenü integriert. Bewegt man den Mauszeiger auf einen Menüpunkt, werden in einem PopUp-Menü die darunterliegenden Menüpunkte (nur direkte Kinder) angezeigt und können ausgewählt werden. Durch direktes Anwählen des Abteilungsunterpunktes im PopUp-Menü überspringt man eine Seite (die Abteilungs-Newsseite) in der Navigation.

4.2 Modularisierung der Templates

Für die Modularisierung der Templates muss zunächst eine Testumgebung geschaffen werden. Da das Ändern eines Templates im CMS mehrere Schritte benötigt und zusätzlich eine gewisse Zeitspanne vom Ändern bis zur Anzeige in der Webausgabe vergeht, stellt das CMS keine geeignete Testumgebung dar.
Microsoft bietet das Tool MSXML[22] zum kostenlosen Download an. Nach der Installation des Tools, kann der Internet-Explorer (ab 5.0) als XSLT-Prozessor verwendet werden. Ein kleines XML/XSL-Beispiel hilft beim Testen. Um die modularisierten Templates testen zu können werden mehrere XML-Dokumente, die viel benutzte Darstellungen repräsentieren, aus dem CMS geladen und lokal gespeichert. Über ein zusätzliches XML-Element (<xml-stylesheet>) kann das zu verwendende XSL-Template festgelegt werden.

Zunächst wird ein Standard-Template programmiert. Die Parameter können beliebig geändert werden, um alle Menüstrukturen und Content-Darstellungen testen zu können. Als Importquelle wird das lokal angelegte Import-Template angegeben.

Als erste Elemente im Import-Template müssen die benötigten Parameter programmiert werden. Aufgrund der umfassenden Layout-Änderung ist es nicht praktikabel, Template-Regeln aus den „alten“ Templates ins Import-Template zu übertragen und abzuändern. Sinnvoller erscheint es das Gerüst der Seite komplett neu zu programmieren und nur die entscheidenden XSL-Anweisungen aus den Template-Regeln der alten Templates zu übernehmen.

Um das Template übersichtlich zu halten wird versucht Template-Regeln zu erstellen, die weniger als 40 Zeilen benötigen. Die erste Regel ist die Master-Regel,

[22] kostenloser Download im Internet; URL: http://www.microsoft.com/downloads/details.aspx?FamilyId=B432CD45-B7ED-4C32-A443-EC56F10EE175&displaylang=en (13.11.2003)

die aus der Start-Regel (Start-Template) aufgerufen wird. Sie bildet den Rahmen der HTML-Datei. Für die Erstellung des HTML-Headers wird eine separate Template-Regel (htmlhead) aufgerufen.
Im Body baut die Regel erste Tabellen für Kopf, linkes Menü und Contentbereich auf.
Die weitere Struktur wird in anderen Template-Regeln, die aus der Master-Regel aufgerufen werden, programmiert. Die Internetseite „SelfHTML“[23] bietet eine sehr gute Hilfe bei der Programmierung der HTML-Anweisungen.

Nachdem das Seitengerüst steht, werden die weiteren Template-Regeln programmiert. Dabei wird von Bereich zu Bereich vorgegangen:

- Seitengerüst
- Kopf mit Abteilungsmenü
- linkes Menü mit QuickSearch und CommunityLogin
- Untermenü (Registermenü, Iconmenü)
- Content-Darstellungen: Standard, News, Exlink, PopUp, AttrTable
- Toolleiste
- Fußnote
- Erstellen der benötigten Style-Anweisungen
- Anpassen der Spezial-Templates und weiterer System-Templates

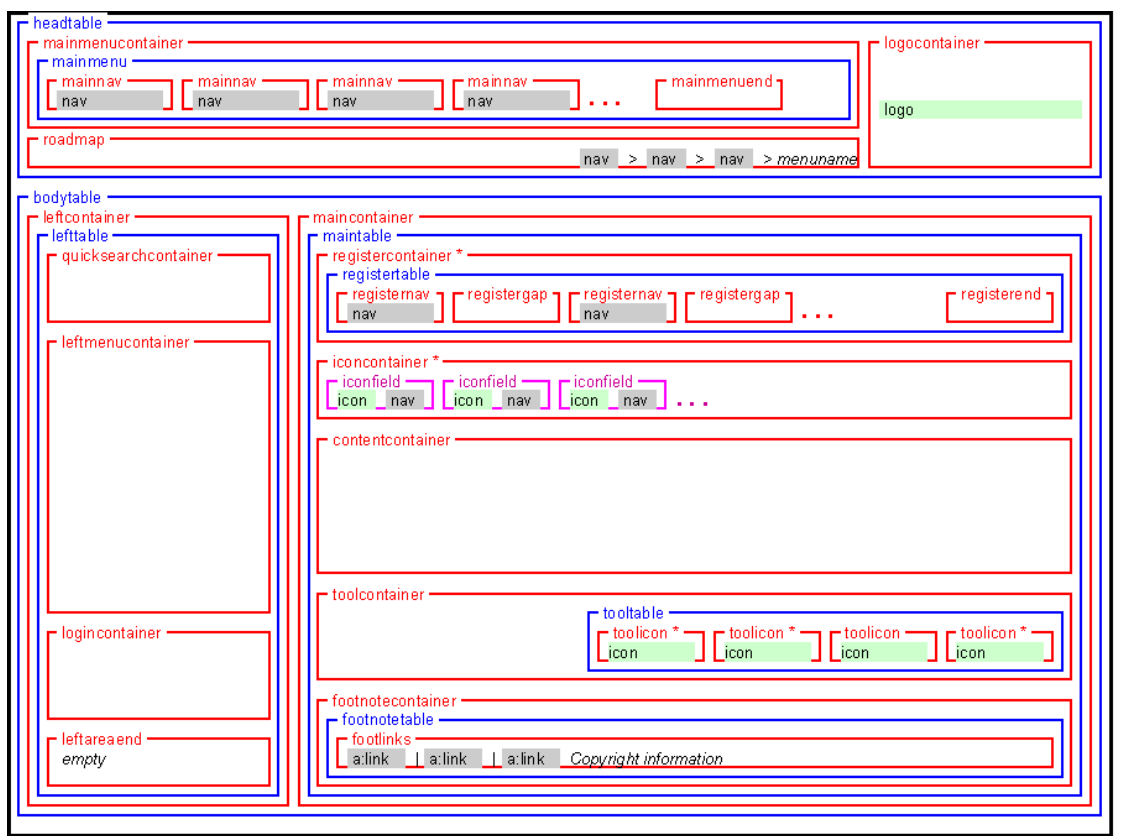


Abbildung 4-2: Tabellen-Layout für eine Intranet-Seite
Quelle: eigene Darstellung

Ein Problem beim Aufbau der Menüs stellt der Wechsel vom linken Menü zum Untermenü dar. Dies funktionierte auch bei den „alten“ Templates nicht immer fehlerfrei. Das Problem besteht darin, dass beim Aufbau des linken Menü nicht

[23] Münz, SelfHTML; Online im Internet; URL: http://selfhtml.teamone.de/ (13.11.2003)

entschieden werden kann, welches die letzte im linken Menü darzustellende Menü-Ebene ist. Bei einem Template vom Typ „icon“ wird der aktuelle Menüpunkt im Iconmenü dargestellt, das Elternmenü im Registermenü. Das Elternmenü des Registermenüs ist folglich das letzte, das im linken Menü dargestellt werden darf. Zu dem Zeitpunkt, an dem dieser Navigationspunkt erstellt wird, ist die Anzahl der Unterebenen aber noch nicht bekannt, wodurch Probleme auftreten können.
Abhilfe schafft ein Parameter (uMenStop), der eine Template-Regel aufruft, die vor Ausführung der Master-Regel die Navigationsstruktur durchläuft und diese Ebene bestimmt. Die Regel durchläuft die geöffneten Menüpunkte, bis der aktuelle Menüpunkt gefunden ist und schreibt, je nach Typ, die Menü-ID des aktuellen Menüs, des Elternmenüs oder des „Urelternmenüs“ in den Parameter.
Die Template-Regel für das linke Menü muss dahingehend geändert werden, dass vor Aufruf der Regel zum Aufbau eines Untermenüs geprüft wird, ob es sich beim gerade erstellten Menüpunkt um den handelt, der in „uMenStop“ festgelegt wurde.

Wie in Kapitel 3.3.2 (S. 24) beschrieben, ist der XSL-Code zur Erzeugung eines Links für einen Menüpunkt in allen Template-Regeln zum Aufbau eines Menüs identisch. Dieses Code-Fragment wird in eine neue Template-Regel (navpoint) programmiert und aus den ursprünglichen Template-Regeln aufgerufen. Die Funktion zur Hervorhebung der geöffneten Navigationspunkte muss in dieser neuen Regel realisiert werden. Dazu werden aus dem XML-Element des Menüpunktes die Attribute „selected“ und „opened“ ausgelesen. Hat eines der Attribute den Wert „true“ wird das ID-Attribut für das Element, in dem der Link erzeugt wird, auf „active“ gesetzt, sonst auf „inactive“.

Nachdem die Template-Regeln für den Menüaufbau programmiert sind werden die Regeln für die verschiedenen Content-Darstellungen programmiert. Neben den Regeln für die verschiedenen Darstellungen, muss eine Regel erstellt werden, in der entschieden wird, welche Regel für die Content-Darstellung auszuführen ist.

Die News-Darstellung soll so gestaltet werden, dass zwei Beiträge nebeneinander platziert werden, wenn mehr als ein Beitrag existiert. Die geplante Sortierung soll die Beiträge von links nach rechts und dann von oben nach unten anordnen. News-Beiträge die jünger als sieben Tage sind, sollen hervorgehoben werden.
Dabei treten mehrere Probleme auf:
Durch die Anordnung von 2 Contents nebeneinander in der Tabellenstruktur, muss vor jedem „ungeraden“ Content ein TR-Element geöffnet werden und nach jedem „geraden“ und nach dem letzten Content das TR-Element geschlossen werden.
Die einfachste Lösung scheint eine entsprechende if-Abfrage vor bzw. nach einem News-Beitrag zu sein:

```
<xsl:if test="position() mod 2">
  <tr>
</xsl:if>
... News Beitrag aufbauen
<xsl:if test="((position()+1) mod 2)or(position()=last())">
  </tr>
</xsl:if>
```

Solch ein Konstrukt, erkennt der XSLT-Prozessor als Fehler, da der Code nicht XML-komform ist. Innerhalb der if-Anweisungen befinden sich keine vollständigen Elemente.

Nach mehreren erfolglosen XSL-Ansätzen scheint eine Lösung des Problems nur mit Hilfe von JavaScript möglich. Anstelle der TableRow-Tags (<tr>) wird ein JavaScript-Element eingefügt, welches die Tags dynamisch im Browser erzeugt.
Der folgende Lösungsweg ohne JavaScript wird erst später realisiert. Die Anweisungen zum Erstellen eines News-Beitrages werden in eine neue Template-Regel „newsbox“ extrahiert. Innerhalb des Table-Elements, in dem die News-Beiträge dargestellt werden, wird die Regel „newsrow“ aufgerufen. Diese baut eine Zeile (2 News-Beiträge) auf. Weitere Zeilen werden durch einen rekursiven Aufruf aufgebaut. Bei der Auswahl des darzustellenden News-Beitrages hilft ein Parameter, der beim Regel-Aufruf übergeben wird. Der folgende Auszug aus dem Quellcode verdeutlicht die Funktion.

```
<xsl:template name="newsrow">
  <xsl:param name="pos">1</xsl:param>
  <tr>
    <xsl:for-each select="/ … /Content">
      … Sortieranweisungen
      <xsl:if test="position() = number($pos)">
        <xsl:call-template name="newsbox"/>
      </xsl:if>
      <xsl:if test="(position() = (number($pos)+1))">
        <xsl:call-template name="newsbox"/>
      </xsl:if>
    </xsl:for-each>
  </tr>
  <xsl:if test="count(/ … /Content)&gt;(number($pos) +1)">
    <xsl:call-template name="newsrow">
      <xsl:with-param name="pos">
        <xsl:value-of select="number($pos)+2"/>
      </xsl:with-param>
    </xsl:call-template>
  </xsl:if>
</xsl:template>
```

Die Sortierung der Beiträge (Contents) kann über XSL-Anweisungen realisiert werden.
Das Erstellungsdatum eines Contents kann aus dem Content-Attribut „CreateDate“ (Attribut-Klasse „Author-Information“) ausgelesen werden. XSL bietet keine Möglichkeit zum Auslesen des aktuellen Datums, daher muss für die Hervorhebung der neuen Beiträge eine JavaScript-Funktion programmiert werden. Als Parameter erhält die Funktion das ausgelesene „CreateDate“ des Contents als Zeichenkette. Die Zeichenkette muss in ein Datumsformat umgewandelt werden um es mit dem aktuellen Datum vergleichen zu können. Ist die Differenz der beiden Daten geringer als sieben Tage, wir für das Table-Element des entsprechenden Beitrags ein ID-Attribut mit dem Wert „new“ erzeugt. Ansonsten erhält das Attribut den Wert „old“.

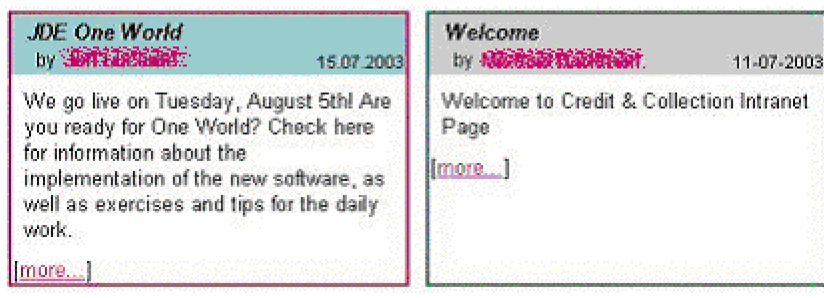


Abbildung 4-3: News-Darstellung von Contents
Quelle: Screenshot einer Intranetseite

Für die Autoren-Informations-Funktion wird ebenfalls JavaScript verwendet. Von jedem Content werden Überschrift, Autor, Email-Adresse des Autors und Erstellungsdatum in ein 2-dimensionales Array geschrieben. Beim Anklicken des Icons für die Funktion wird im Javascript ein HTML-String erzeugt, der die Autoren-Informationen beinhaltet. Dieser String wird in ein neu geöffnetes Fenster (PopUp) geschrieben.

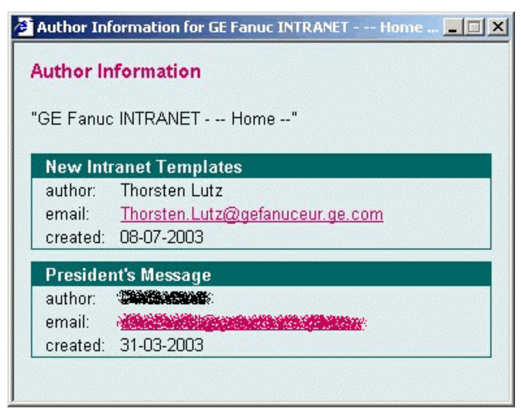


Abbildung 4-4: Autoren-Informationen
Quelle: Screenshot eines Browserfensters

Die Realisierung der PopUp-Menüs wirft zusätzliche Probleme auf. Um die Menüs einzeln sichtbar und wieder unsichtbar machen zu können muss jedes Menü in ein Block-Element (div) eingefasst sein. Die Blockelemente können durch Wertänderung des Style-Attributes „visibility“ ein- und ausgeblendet werden. Die eigentliche Intranet-Seite muss ebenfalls in ein Blockelement eingefasst werden. Die Blockelemente für die PopUp-Menüs werden mit Hilfe von JavaScript-Funktionen erzeugt. Aufgerufen werden die Funktionen aus der Master-Regel, da sie auf einer Ebene mit der eigentlichen Intranet-Seite stehen. In der Template-Regel zum Aufbau der Navigationspunkte im Abteilungsmenü muss ein JavaScript-Aufruf eingefügt werden. Die aufgerufene Funktion muss beim „Überfahren“ eines Menüpunktes mit der Maus das entsprechende Blockelement mit dem Untermenü sichtbar machen. In die Template-Regeln, die die umliegenden Bereiche aufbauen, muss ebenfalls ein JavaScript-Aufruf implementiert werden. Dieser muss bei beim „Überfahren“ der umliegenden Bereiche mit der Maus das PopUp-Menü ausblenden.

Nachdem das Import-Template fertiggestellt ist können verschiedene Darstellungsformen aus den gespeicherten XML-Dateien generiert werden. Weil die CSS noch nicht programmiert ist, bietet die Darstellung keinerlei Designanpassungen. Die Tabellenstruktur lässt sich sehr gut nachvollziehen. Im nächsten Schritt wird die CSS programmiert. Als Hilfe erweist sich dabei die Webseite „css4you“[24], die eine sehr gute Übersicht über existierende Style-Anweisungen und deren Browser-Unterstützung bietet. Die Zuweisung der verschiedenen Styles geschieht über die in den Tabellenzellen vergebenen Klassen. Das Design wird weitestgehend von den Mitarbeitern des eBusiness-Teams vorgegeben.

[24] Online im Internet; URL: http://www.css4you.de/ (13.11.2003)

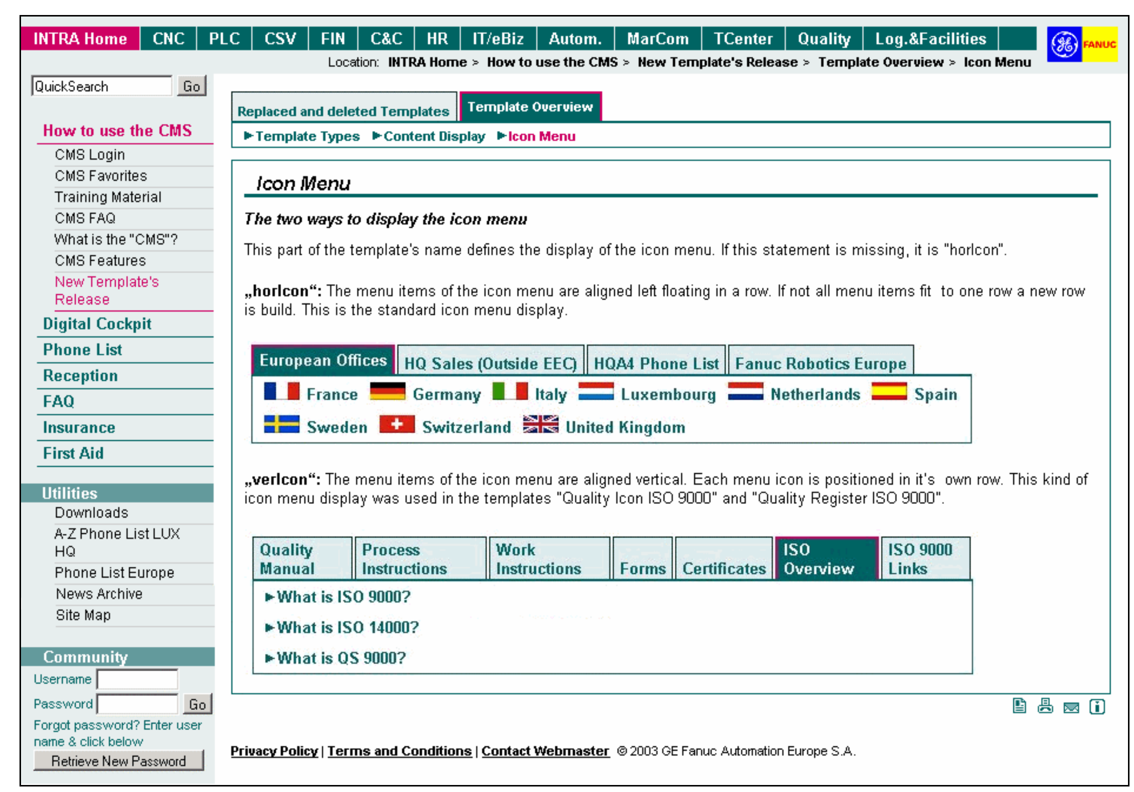


Abbildung 4-5: Intranet-Seite nach der Modularisierung
Quelle: Screenshot einer Intranetseite

Nachdem Standard-Template und Import-Template fertig sind, werden die Spezial-Templates auf das neue Konzept umgestellt. Anschließend werden die System-Templates „PrintView“ und „PlainContentWithoutNavigation“ angepasst.

Durch die Modularisierung der Templates wurde ihre Anzahl stark reduziert. Aus 73 verschiedenen Templates wurden neun Standard-Templates, zwei System-Templates und sechs Spezial-Templates, eine Reduzierung um mehr als 70%. Durch neue Funktionen und Anforderungen wurden drei weitere Templates erstellt.
Das Import-Template ist von ca. 150 Zeilen auf ca. 1100 Zeilen angewachsen, enthält 38 Template-Regeln, statt bisher drei. Die Templates sind nicht nur für Intranet-, sondern auch für Internet- und Extranet-Anwendungen verwendbar. Außerdem wurden einige zusätzliche Funktionen integriert und eine ausführliche Kommentierung vorgenommen.
Die Standard-Templates, die vorher im Durchschnitt aus ca. 750 Zeilen bestanden wurden auf eine Größe von 20 Zeilen verringert. Auch bei den Spezial-Templates konnte die Größe erheblich reduziert werden..

Alle Design-Änderungen können nun unabhängig von CMS und XSL durchgeführt werden. Änderungen im Layout oder in der Template-Funktion müssen nur noch an einer einzigen Stelle im Import-Template durchgeführt werden. Die Pflege der Start-Templates ist aufgrund der geringen Größe wesentlich leichter geworden.

4.3 Dokumentation für Redakteure und Anwender

Nach der Überarbeitung der Templates müssen die Benutzer über die anstehenden Änderungen informiert werden. Dies wird über das CMS geschehen. Unter dem Menüpunkt „How to use the CMS“ wird ein News-Beitrag erstellt, der über die

Umstellung informiert und Links zu den unterstützenden Intranet-Seiten enthält. Zusätzlich erhält der Menüpunkt zwei neue Unterpunkte. Unter „CMS-Features" werden neue und bereits bestehende Funktionen beschrieben. Der Punkt ist unterteilt in weitere Menüpunkte, die sich detailliert mit den Funktionen befassen. Der Menüpunkt „new Template's Release" richtet sich speziell an die CMS-Redakteure. Dieser Menüpunkt und seine Unterpunkte stellen die neuen Templates vor. Der Unterpunkt „Replaced and deleted Templates" zeigt, welche Templates umbenannt wurden und welche durch andere Templates ersetzt wurden. Der Unterpunkt „Template Overview" bietet eine Übersicht über die existierenden Templates und beschreibt, gegliedert nach den Parametern, deren Menü- und Content-Darstellung.
Auf der Intranet-Startseite wird ebenfalls ein News-Beitrag platziert, der auf die Änderung hinweist.
Die neuen Menüpunkte und Contents werden publiziert, wenn der Relaunch der Templates beendet ist.
Screenshots der Dokumentation können im Anhang (S. 56) betrachtet werden.

4.4 Relaunch der neuen Templates

Da GE Fanuc zwei parallele CMS-Systeme (Test- und Produktiv-System) betreibt werden die überarbeiteten Templates zunächst auf dem Testsystem installiert. Diese Installation dient als Test für die Umstellung des Produktivsystems und soll Probleme aufdecken, damit der Relaunch des Produktivsystems zügig und fehlerfrei verläuft.

Vor der Umschaltung musste geklärt werden in welche Verzeichnisse die benötigten externen Dateien (CSS, JavaScript und einige .gif-Icons) hinterlegt werden und wie die „alten" Templatezuordnungen auf die neuen umgeschaltet werden.

Für die externen Dateien wurde ein Verzeichnis auf dem CMS-Server festgelegt.

Die Templatezuordnung zu einem Navigationspunkt wird vom CMS in einer Datenbanktabelle anhand der Template-ID (uTem) durchgeführt. Anstatt den gesamten Navigationsbaum der Intranetseite zu durchlaufen und die Templatezuordnung zu korrigieren, wird eine Ersetzung der „uTem" der alten Templates durch die der neuen Templates mit Hilfe eines SQL-Statements in der Datenbank vorgesehen.
Als Vorbereitung dazu werden die Template-ID's aller betroffenen Templates gelistet.

Es wird eine Liste mit den durchzuführenden Arbeitsschritten aufgestellt. Während des Relaunch des Test-Systems werden die benötigten Zeiten für jeden Arbeitsschritt, sowie auftretende Probleme festgehalten. Die verwendeten SQL-Statements werden gesichert, um sie beim Relaunch des Produktiv-Systems zu verwenden.

Der Relaunch kann in 3 Phasen unterteilt werden. Während der Umschaltung, steht die Webausgabe des CMS nicht zur Verfügung. Diese Phase muss so schnell wie möglich abgearbeitet werden, da die Aufgaben vieler Mitarbeiter vom Intranet abhängig sind. Die im Testlauf festgestellten Zeiten sind in Klammern angegeben.

- **Vorbereitung**
  1. Attribut-Klasse EXlink erweitern (10 min.)
  2. benötigte Grafiken in Ordner systemstatic/systemimages kopieren (2 min.)
  3. JavaScript-Datei scripts.js nach systemstatic/addin kopieren (1 min.)
  4. neue Templates in CMS übertragen. (45 min.)
  5. Template-Namen anpassen (10 min.)
  6. zusätzliche HOW TO – Menüpunkte freischalten (2 min.)

- **Umschaltung**
  7. Templates publizieren (10 min.)
  8. CSS-Datei styleintranet.css durch neue Version ersetzen. (3 min.)
  9. Menüpunkte mit ersetzten Templates in SQL-DB umschalten (15 min.)

- **Nacharbeiten**
  10. Veraltete Templates löschen (5 min.)
  11. Attribute der Klasse EXlink für betroffene Contents anpassen

Nach dem Relaunch des Testsystems fällt auf, dass einige Seiten nicht dargestellt werden können. Der Grund dafür sind Templates, die bei der Listung nicht aufgeführt wurden (siehe Kapitel 3.3, S. 19), da sie in einem „falschen“ Ordner abgelegt waren und somit nicht bei der Modularisierung berücksichtigt wurden. Es handelt sich um Templates, die eine externe Grafik, die im Content-Attribut „ImageURL“ der Attribut-Klasse „ExternalGraphics“ referenziert ist, in den Contentbereich einfügt. Das Problem wird gelöst, indem eine neue Template-Regel für diesen Fall ins Import-Template implementiert wird. Um zu bestimmen, ob diese Template-Regel ausgeführt werden muss, wird bei jedem Content überprüft, ob das Attribut „ImageURL“ existiert. Die Template-ID’s dieser Templates müssen in die SQL-Statements zur Ersetzung in der Datenbank eingepflegt werden.

5 Entwicklung eines Tools zur Verwaltung und Generierung von Templates

Die abschließende Aufgabe dieser Arbeit stellt die Entwicklung eines Tools dar, mit dem sich die existierenden Templates verwalten und neue Templates generieren lassen.

Die anfängliche Idee bestand darin, ein Programm zu entwickeln, mit dem ohne XSL-Kenntnisse neue Templates, durch Auswahl verschiedener XSL-Komponenten (Template-Regeln), ähnlich einem Baukasten-System, zu erzeugen.

Durch die eingehende Analyse der XSL-Templates und die daraus resultierende Modularisierung wurde der Bedarf nach solchen individuellen Templates auf ein Minimum reduziert. Durch die Einführung der Parameter in den Start-Templates, kann sehr leicht eine individuelle Anpassung vorgenommen werden. Die Funktion, für die das Tool anfänglich gedacht war, wird zukünftig selten benötigt werden.

Als Name des Tools wird TεToo, abgeleitet aus Template-Tool, gewählt. Das phonetische Symbol ε fördert die korrekte Aussprache. Die Assoziation zum englischen Wort Tattoo (Tätowierung) ist bewusst gewählt, da Templates und Tätowierungen im weiteren Sinne mit Layout und Design arbeiten.

5.1 Erarbeitung eines Konzepts

Im Anschluss an die Überarbeitung der Templates, wurde ein neues Konzept für das Tool entwickelt, das auf dem neuen Template-Konzept aufbaut. Im Mittelpunkt steht nicht mehr das Erstellen eines individuellen Templates, sondern die Verwaltung der existierenden Templates. Dabei wird die bevorstehende Realisierung eines CMS-basierten Extranets, die Umstellung der Haupt-Internetseite (www.gefanuc-europe.com) auf das CMS und die Portierbarkeit der Templates in andere Webs berücksichtigt.
Grundgedanke des neuen Konzeptes ist die Verwaltung aller Templates, die für ein Web (z.B. Intranet) benötigt werden, als feste Einheit. Die Templates bilden ein Template-Set im Rahmen eines Projektes. Als Ausgangsbasis dient das Template-Set des Intranet. Andere für die Darstellung eines Webauftrittes wichtige Dateien (CSS und JavaScript) werden durch das Tool verwaltet und gehören zum Projekt.
Den Kern eines Projektes bildet die Projektdatei, die Informationen über das Projekt und die zu einem Projekt gehörenden Dateien beinhaltet.

Wie bei der Modularisierung werden die Templates in Standard-, Spezial- und System-Templates unterteilt. Da in allen Start-Templates (Standard- und Spezial-Templates) über die gleichen Parameter (mit verschiedenen Werten) deklariert und die gleichen Importquellen angegeben sind liegt es nahe diese Informationen in der Projektdatei zu verwalten. Die verschiedenen Komponenten eines Projekts werden in verschiedenen Klassen dargestellt.

5.1.1 Funktionen des Tools

Da das „Template-Set“ eines Webs als Projekt behandelt wird, ist es nötig, neue Projekte anlegen und bestehende Projekte öffnen zu können um damit zu arbeiten. Nach der Arbeit muss ein Projekt abgespeichert werden können.
Um ein einfaches Arbeiten zu ermöglichen wird es eine grafische Oberfläche geben, die neben einer Menüleiste aus dem Projektfenster besteht. Das Projektfenster beinhaltet eine Registeransicht. Es wird 5 Register geben: „Parameter“, „Standard-Templates“, „Special-Templates“, „System-Templates“ und „CSS“.
Aufgrund des Modularisierungs-Konzeptes spielen die Parameter bei Start-Templates (Standard- und Spezial-Templates) eine wichtige Rolle. Ein Hinzufügen, Ändern oder Löschen eines Parameters wird in allen Start-Templates erfolgen. Dies gilt nicht für die Werte eines Parameters. Diese werden von Template zu Template individuell änderbar sein. Auch die Angabe von Import-Quellen, die sich ebenfalls hinzufügen und entfernen lassen, wird in allen Start-Templates gleich sein. Das Register „Parameter“ wird eine Übersicht über die vorhandenen Parameter und Importquellen bieten. Über Buttons im Register lassen sich Änderungen durchführen.

- In **Standard-Templates** wird man nur Werte der Standard-Parameter ändern können. Standard-Templates lassen sich hinzufügen und entfernen. Es wird eine Prüfung geben, die das Erstellen identischer Standard-Templates verhindert. Es wird möglich sein, die Start-Regel zu editieren. Dies wird Auswirkungen auf alle Standard- und Spezial-Templates haben. Eine Übersicht über die existierenden Standard-Templates stellt das Register „Standard-Templates“ bereit. Man hat die Möglichkeit ein einzelnes Template auszuwählen. Buttons im Register bieten Möglichkeiten zur Bearbeitung.
- **Spezial-Templates** sind alle Start-Templates, bei denen ein Spezial-Parameter einen Wert hat oder bei dem neben der Start-Regel weitere Template-Regeln existieren. Ein Editieren der Start-Regel wird sich auf alle Start-Templates auswirken. Alle anderen Änderungen beziehen sich nur auf das aktuelle Template. Das Register „Special-Templates“ zeigt existierende Spezial-Templates an und bietet die Möglichkeit ein Template anzuwählen. Die Bearbeitung erfolgt über Buttons und den Menüpunkt „Template“.
- **System-Templates** haben einen Aufbau, der sich von dem der Start-Templates stark unterscheidet. Sie lassen sich individuell ändern und werden im Register „System-Templates“ angezeigt und lassen sich wie die Start-Templates auswählen und bearbeiten.

Das Register „cssTab“ bietet eine Übersicht über die dem Web zugeordneten CSS und JavaScripte (JS). Diese Dateien können durch das Tool umbenannt oder gelöscht werden. Ein Hinzufügen neuer Dateien ist möglich. Zum Bearbeiten der Dateien öffnet das Tool externe Anwendungen, die durch den Benutzer ausgewählt werden. Das Einfügen eines entsprechenden <link>- bzw. <script>-Elements in den Quellcode eines Templates wäre möglich. Dazu benötigt man die Angabe des relativen Pfades auf dem Webserver. Außerdem muss die Einfügeposition für ein entsprechendes Element im Quellcode gesucht werden. Diese Funktion ist vorerst nicht geplant, kann jedoch nachträglich realisiert werden.
Hat man in den Registern „Standard-Templates“. „Special-Templates“ oder „System-Templates“ ein Template ausgewählt, hat man die Möglichkeit Template-Regeln zu editieren, neu anzulegen oder zu löschen.

Eine detailliertere Beschreibung der Funktionen und der Handhabung des Tools bietet das Benutzerhandbuch im Anhang (Kapitel 7.6, S. 64).

5.1.2 Anpassung der Templates

Damit das Tool korrekt arbeiten kann, müssen bestimmte Bedingungen in einem Template-Set erfüllt sein. Jedes Template muss daher nach einem bestimmten Muster aufgebaut sein.
Die ersten Knoten unter der Dokument-Wurzel sind Kommentarknoten. Sie enthalten Name, Version, Beschreibung und Änderungsdatum des Templates und eine Angabe über das Projekt (Web) zu dem das Template gehört. Die Reihenfolge spielt keine Rolle. Ist einer dieser Knoten nicht vorhanden, wird er vom Tool angelegt.
Die Importangaben sucht das Tool eigenständig und fügt neue Import-Angaben selbstständig in das Template ein. Auskommentierte Importangaben werden ebenfalls berücksichtigt, sofern diese Kommentar-Knoten vor dem `<xsl:output>`-Knoten stehen.
Nach dem `<xsl:output>`-Knoten folgen die Parameter. Standard- und Spezial-Parameter können im Template nicht unterschieden werden. Sie werden nur innerhalb des Tools in der Projektdatei unterschieden. Ein Parameter besteht zwingend aus dem `<xsl:param>`-Element und einem optionalen Kommentarknoten, der als Beschreibung des Parameters dient. Parameter von System-Templates werden durch das Tool ausgelesen. Die Start-Regel von Start-Templates muss einen bestimmten Namen haben, der im Tool festgelegt wird.

Das Syntaxdiagramm für XSL-Templates und die Projektdatei kann im Benutzerhandbuch im Anhang nachgeschlagen werden..

5.1.3 Technische Aspekte

Für die Verwaltung der Templates werden 2 Personen (eBusiness-Team) verantwortlich sein. Der Speicherort für die Template-Sets wird ein Server-Laufwerk sein, auf das beide Personen Zugriff haben. Eine browserbasierte Realisierung ist daher nicht erforderlich.

Die Laufzeit des Programms spielt eine untergeordnete Rolle, da keine zeitkritischen Aktionen mit Hilfe des Programms vorgenommen werden. Daher wird für die Realisierung des Tools die Programmiersprache Java verwendet. Weitere Vorteile die für den Einsatz von Java sprechen sind die für Java 1.4 angebotenen Klassen-Bibliotheken zum Umgang mit XML und DOM, sowie zur Erstellung grafischer Benutzeroberflächen.

Zur Speicherung der Template-Sets gibt es mehrere Alternativen:

- Ein direktes Arbeiten auf der CMS-Datenbank hat den Vorteil, dass das umständliche Ändern eines Templates im CMS-Admin-Tool entfällt. Für das Anlegen eines neuen Templates oder eines neuen Template-Sets müssen viele Operationen auf verschiedenen Tabellen der Datenbank ausgeführt werden. Für die ordnungsgemäße Funktion der CMS-Datenbank ist IC Team GmbH verantwortlich. Nur IC Content 4.0 arbeitet auf der Datenbank. Durch ein manipulieren der Datenbank durch das Tool wird es schwieriger bei Datenbankfehler Ansprüche gegen IC Team GmbH durchzusetzen. Für die

Verwaltung von JS-Dateien und CSS ist die CMS-Datenbank zudem nicht vorgesehen.

- Die Verwaltung der Templates innerhalb eines Java-Objekts hat den Vorteil, dass man das ganze Projekt, evtl. sogar mehrere Projekte in einer einzigen Datei verwaltet. Es muss eine Import-Export-Funktion existieren, um eine XSL- ,CSS-, oder JS-Datei zu erzeugen oder diese ins Projekt zu integrieren. Da ein Template manuell ins CMS eingepflegt werden muss, muss nach einer Änderung meist jedes Template exportiert werden. Um Dateien mit anderen Programmen zu bearbeiten, muss die entsprechende Datei ebenfalls exportiert werden.
- Das Speichern der Dateien in einem Projektordner als „gewöhnliche" Dateien erscheint aufgrund der Anforderungen als geeignetste Alternative und wurde zur Realisierung gewählt. Als Hilfsmittel dient eine Projekt-Datei, die die wesentlichen Daten des Projektes als XML-Code enthält.

Da 2 Personen für die Verwaltung der Templates verantwortlich sind, wird ein Sicherungsmechanismus implementiert, damit ein gleichzeitiges Bearbeiten eines Projektes durch mehrere Personen unterbunden wird.

Die Klassen des Tools werden in drei Pakete (Packages) unterteilt. Das erste Paket „templateTool" beinhaltet die Klassen, die die wichtigsten Funktionen des Tools bereitstellen. Das Paket „templateTool.project" enthält die Klassen, die das Projekt repräsentieren. Die Klassen enthalten Methoden zum Ändern von Projekteigenschaften. Das dritte Package („templateTool.gui") beinhaltet Klassen, die grafische Oberflächen erstellen.

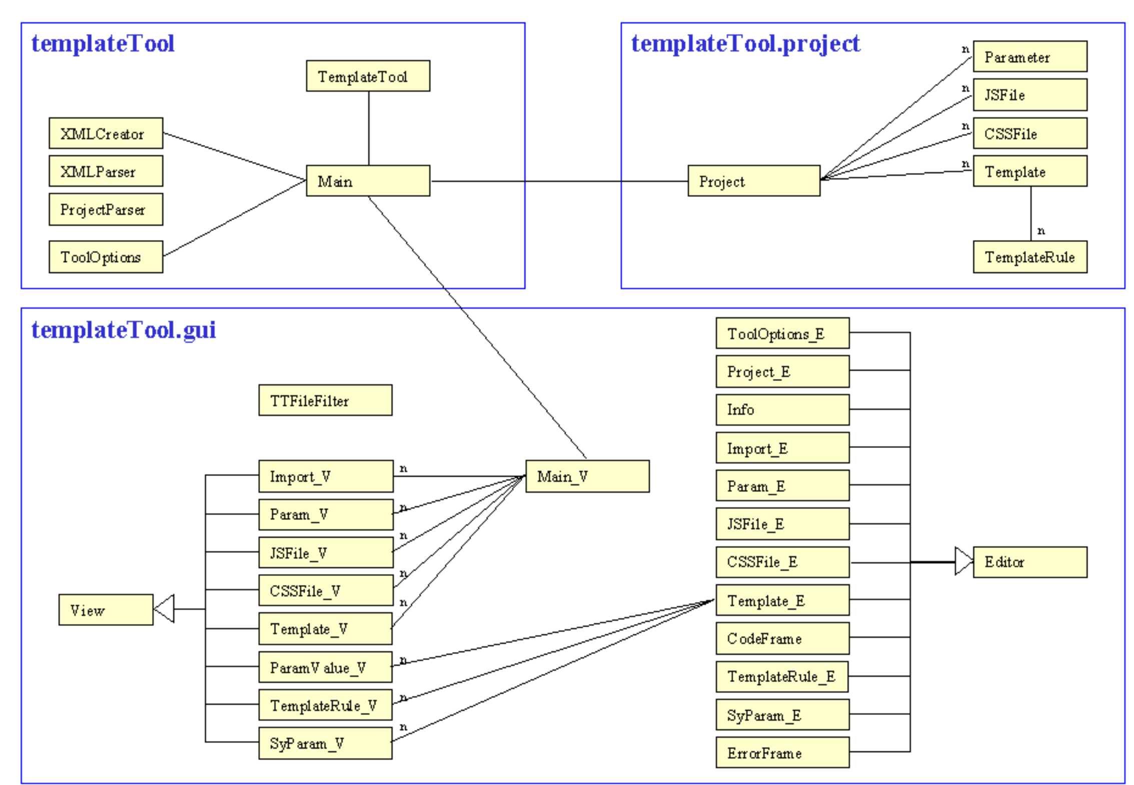


Abbildung 5-1: Package-Diagramm
Quelle: eigene Darstellung

5.2 Realisierung

Die Java 2 Standard Edition J2SE v 1.4.2_02[25] (Java 2), die bei Sun[26] zum kostenlosen Download bereitsteht dient als Entwicklungsplattform. Als wichtige Hilfe dient die Online API-Spezifikation[27]. Sie umfasst eine detaillierte Beschreibung aller in der J2SE bereitgestellten Klassenbibliotheken, deren Klassen und Methoden.

Die ersten Klassen die implementiert werden sind die Klassen des Paketes „templateTool.project“ (Projekt-Klassen). Die einzelnen Klassen stellen die durch das Tool änderbaren Komponenten eines Projektes dar. Sie erhalten Methoden zum Anlegen, Verändern und Löschen einzelner Komponenten. Jede der Komponenten enthält Variablen zur Aufnahme von Meta-Informationen, die im Template als Kommentare eingefügt werden. Ein Projekt besteht aus einem Objekt der Klasse „Project“, dass Objekte der weiteren Projekt-Klassen enthält.

Anschließend werden die Klassen TemplateTool und Main implementiert. TemplateTool beinhaltet die Main-Methode, in der ein Objekt der Klasse Main erzeugt und ausgeführt wird. Als Argumente können Projektdatei und Benutzername übergeben werden. Während der Entwicklung des Tools werden durch Änderung der Main-Methode verschiedene Test-Situationen erzeugt, um neu implementierte Klassen zu testen.
Die Klasse Main bildet den Kern von TεToo. Sie koordiniert das Zusammenspiel aller Klassen und verknüpft das Projekt mit der grafischen Benutzeroberfläche.

Das Auslesen der Projektdatei und der Templates wird mit Hilfe eines XML-Parsers realisiert. Die implementierte Klasse „XML-Parser“ öffnet im Dateisystem gespeicherte XML-Dateien und erzeugt aus dem XML-Code einen DOM[28]-Baum. Aus den nach dem Parsen entstandenen DOM-Bäumen werden die Instanzen der Projekt-Klassen (Projekt-Objekte) generiert. Bei Templates wird das in der Methode „analyse_Template“ in der Klasse XML-Parser realisiert. Für die Projektdatei stellt die Klasse „ProjectParser“ die benötigten Methoden zur Verfügung. Für das Parsen und die Bearbeitung der entstehenden DOM-Bäume werden die standardmäßig in J2SE enthaltenen Klassenbibliotheken „javax.xml“, „org.w3c.dom“, „org.xml.sax“ verwendet.

Nachdem die Klassen und Methoden zum Öffnen eines Projektes implementiert sind, werden die Klassen und Methoden zur Bearbeitung des Projektes implementiert. Dazu zählen auch grafische Oberflächen (GUI).
In den Registern des Hauptfensters werden die Daten der Projekt-Objekte in Form von View-Objekten dargestellt. Ein View-Objekt besteht aus einem oder mehreren Textfeldern, die die Daten anzeigen und einem „Edit“-Button, der zum Bearbeiten des angezeigten Objekts ein neues Fenster (Editor) öffnet. Für jede Projekt-Klasse gibt es eine View-Klasse. Gemeinsame Funktionen der View-Klassen werden in einer Oberklasse („View“) implementiert.

---

[25] Download im Internet; URL: http://java.sun.com/j2se/1.4.2/download.html (13.11.2003)
[26] Sun Microsystems; Online im Internet; URL: http://www.sun.com/ (13.11.2003)
[27] Online im Internet;URL: http://java.sun.com/j2se/1.4.2/docs/api/index.html (13.11.2003)
[28] DOM: **D**ocument **O**bject **M**odel; Online im Internet; URL: http://www.w3.org/DOM/ (13.11.2003)

Aufgrund mehrerer gleicher Funktionen in den verschiedenen Editoren, (Übernahme oder Verwerfung von Änderungen, Fenster schließen usw.) wird eine Oberklasse „Editor“ erstellt, die die gemeinsamen Funktionen zur Verfügung stellt. Die einzelnen Editoren erweitern die Oberklasse für die jeweiligen Projekt-Klassen.

Bevor Änderungen, die in Editoren vorgenommen werden, gespeichert werden, wird eine Prüffunktion aufgerufen, die alle Werte auf Gültigkeit prüft. Bei Template-Regeln und System-Parametern muss der Code auf XML-Konformität geprüft werden. Dazu wird die Klasse „XML-Parser“ so erweitert, dass auch Zeichenketten geparst werden können. Ein Vorteil der verwendeten Klassenbibliotheken sind recht genaue Fehlerbeschreibungen bei Parsing-Fehlern. Die Prüffunktionen für andere Daten werden zentral in die Klasse Main integriert.

Zum Generieren der Projektdatei und der Template-Dateien wird je eine Methode in die Klasse „XMLCreator“ programmiert. Die Arbeitsweise beider Methoden ist gleich. Die für die Datei benötigten Daten werden aus den Projekt-Objekten ausgelesen. Es wird ein DOM-Baum erstellt der die Knoten der zu erzeugenden XML-Datei enthält. Der DOM-Baum wird, gemeinsam mit dem Dateinamen, an eine Methode zum Speichern übergeben. Er wird in einen Stream umgewandelt, der in die angegebene Datei geschrieben wird. Die übliche Einrückung bei XML-Dateien, die für Übersicht sorgt, wird bei der Umwandlung nicht hergestellt. Über zusätzliche an den gewünschten Stellen in den DOM-Baum eingefügte Textknoten, die Tabulatoren oder Zeilenumbrüche enthalten, wird eine Einrückung beim Speichern erreicht. Ein Parser, der eine so erzeugte Datei parst, ignoriert Tabulatoren und Zeilenumbrüche, so dass diese Lösung keine negativen Auswirkungen auf die Funktion der Datei hat.

Standardmäßig werden XML-Dokumente in „UTF-8“[29] kodiert. Dabei tritt ein Problem mit Sonderzeichen im Code auf.
Bsp.: Ein „Ü“ wird in HTML „Ü“ geschrieben. Beim Öffnen einer in UTF-8 kodierten Datei mit dem Tool wird beim Parsen ein „Ü“ erzeugt. Beim Versuch die Datei noch einmal zu speichern, tritt ein Fehler bei der Umwandlung des DOM-Baums auf, da das Zeichen „Ü“ nicht zulässig ist.
Eine andere Kodierung (z.B. ISO 8859-1[30]) könnte das Problem lösen. Es muss sichergestellt werden, dass der XSLT-Prozessor des CMS die Kodierung akzeptiert. Dazu wird im CMS ein „ISO 8859-1“-kodiertes Test-Template erzeugt und ausgeführt über einen Testmenüpunkt ausgeführt. Der Test verläuft positiv. Die neue Kodierung muss beim Schreiben und beim Lesen der Datei eingestellt werden.

Während der Laufzeit des Programms können zwei Arten von Fehlern auftreten: Fehler durch Falscheingaben des Benutzers und von der Java-Laufzeitumgebung (JRE) erzeugte Ausnahmefehler (Exceptions). Für die Anzeige von Fehlermeldungen wird eine Klasse „ErrorFrame“ erzeugt, die sich aus der Klasse Editor ableitet. Die Klasse erstellt eine GUI in der eine Fehlerbeschreibung angezeigt wird. Bei Ausnahmefehlern wird zusätzlich der StackTrace angezeigt, mit dem sich der Ursprung des Fehlers feststellen lässt.

[29] UTF-8: Unicode Transformation Format-8; Online im Internet;
URL: http://www.cl.cam.ac.uk/~mgk25/unicode.html#utf-8 (13.11.2003)

[30] Vgl. Münz, SelfHTML;
URL: http://selfhtml.teamone.de/inter/zeichensaetze.htm#iso8859_liste (13.11.2003)

Die Implementierung des Sicherungsmechanismus erweist sich als problematisch. Da alle relevanten Daten als Dateien vorliegen und erst vom Tool in Java-Objekte umgewandelt werden, gibt es kein Programmierkonstrukt, das hundertprozentige Sicherheit garantiert. Eine relativ sichere Lösung ist die Verwendung eines „File-Locks“. Dabei wird sofort nach Öffnen des Projektes eine Datei (Lock-File) in den Dateiordner des Projektes geschrieben. Nach dem Schließen des Projektes wird das Lock-File gelöscht. Vor dem Öffnen eines Projektes prüft das Programm, ob in dem Dateiordner ein Lock-File existiert. Ist dies der Fall, darf das Projekt nicht geöffnet werden. Dieser Sicherungsmechanismus bietet eine kurze Zeitspanne, in der ein Öffnen durch mehrere Benutzer möglich ist. Für zwei bis drei Benutzer stellt ein File-Lock eine ausreichende Sicherheitsmaßnahme dar..
Der File-Lock des Tools wird so implementiert, dass das Lock-File Informationen über den Benutzer des Projektes und den Zeitpunkt der Sperrung beinhaltet. Dadurch kann eine genauere Meldung bei einem gesperrten Projekt erzeugt werden. Die Sperre wird nicht auf das Projekt, sondern auf den Ordner gelegt, in dem die Projektdatei sich befindet. Denkbar wäre, dass in einem Ordner 2 Projekte existieren, die XSL-Templates mit gleichem Namen erzeugen. In diesem Fall muss verhindert werden, dass beide Projekte gleichzeitig geöffnet werden. Durch die Sperrung des gesamten Ordners wird das gewährleistet.

Das entwickelte Tool besteht aus 36 Klassen, die in 3 Pakete aufgeteilt sind. Insgesamt umfasst der Quellcode ca. 7200 Zeilen. Mit Hilfe des in J2SE enthaltene Tool Javadoc wird eine umfassende Online-API für das Projekt erstellt. Die erzeugten Klassen werden gemeinsam mit einigen Zusatzdateien (u.a. Batch-Datei zum Start des Tools und Grafik mit Logo des Tools) in ein JAR-Archiv zusammengefasst. Bei installierter Java-Laufzeitumgebung kann man das Tool unter Windows durch einen Doppelklick auf die JAR-Datei starten. Ein weiterer Vorteil des JAR-Archivs besteht darin, dass im Dateisystem nur eine Datei angezeigt wird.

5.3 Testphase

Während der gesamten Entwicklung des Tools werden Tests durchgeführt, um die ordnungsgemäße Funktion der implementierten Klassen und Methoden zu prüfen. Zum Testen der Funktion wird manuell eine Projektdatei erstellt. Die Ausgabe der Daten auf der Command-Shell zeigt, ob eine Funktion ordnungsgemäß arbeitet.

Funktionstest
Abschließend wird eine Testreihe durchgeführt, in der alle Funktionen des Tools geprüft werden. Neben der eigentlichen Funktion wird auch das Verhalten im Fehlerfall weitestgehend geprüft. Einige Situationen die zu einem Ausnahmefehler führen können nicht „künstlich“ erzeugt werden.

Die nachfolgende Übersicht zeigt die wichtigsten Testfälle.

Funktion	**Testfall**	**Beschreibung**
Projekt öffnen	korrektes Projekt	Projektdatei ist korrekt und alle Dateien (CSS, JS und Templates) sind vorhanden
	Nicht XML-konforme Projektdatei	Die zu öffnende Projektdatei ist nicht XML-konform
	Fehlende Datei	Die Projektdatei verweist auf eine nicht existente Datei (CSS, JS und Template)
	Nicht XML-konformes Template	Der Code eines Templates ist nicht XML-konform
	FileLock aktiviert	Im Verzeichnis des zu öffnenden Projektes existiert ein Lock-File
Projekt erstellen	Projekt überschreiben	Es wird versucht eine Projekt anzulegen, dass bereits im Projektpfad existiert
	Projekt in neuem Projektpfad erstellen	Der angegebene Projektpfad existiert noch nicht
	Eingabefehler	In den Eingabefeldern des Projekt-Editors werden ungültige Werte eingetragen
Hauptfenster	„add Template“	Ein neues Template soll generiert werden
	„import Template“	Ein vorhandenes Template soll in das Projekt integriert werden
	„add Import“	Es soll eine neue Importquelle angelegt werden
	„add Parameter“	Es wird versucht einen neuen Parameter zu erzeugen
	„add CSS“	Eine bestehende CSS-Datei soll ins Projekt integriert werden
	„add JavaScript“	Eine bestehende JavaScript-Datei soll ins Projekt integriert werden
	Tool-Einstellungen	Die Tool-Einstellungen werden geändert
Editoren (allgemein.)	„OK“ ohne Fehler	Die Eingabefelder der Editoren enthalten gültige Werte
	„OK“ mit Fehlern	Je ein Eingabefeld enthält einen unzulässigen Wert
	„Save“ ohne Fehler	Die Eingabefelder der Editoren enthalten gültige Werte
	„Save“ mit Fehlern	Je ein Eingabefeld enthält einen unzulässigen Wert
	„Cancel”	Die Eingaben seit dem letzten „Save“ sollen verworfen werden
	Editorfenster schließen	Die Eingaben seit dem letzten „Save“ sollen verworfen werden
Parameter-Editor	“Possible values” testen	Verschiedene Wertelisten werden eingetragen und deren Interpretation überprüft
	“Value for all start-templates” testen	Der angegebene Wert muss in allen Start-Templates erscheinen

Template-Rule-Editor	Korrekter Code	Der eingetragene XSL-Code ist XML-konform
	Nicht XML-konformer Code	Der eingetragene XSL-Code enthält nicht XML-konforme Elemente
System-Parameter-Editor	Korrekter Code	Der eingetragene XSL-Code ist XML-konform
	Nicht XML-konformer Code	Der eingetragene XSL-Code enthält nicht XML-konforme Elemente
Template-Editor	„show Code“	Der gesamte XSL-Code des Templates soll in einem Fenster angezeigt werden
	“add Template-Rule”	Im Template soll eine neue Template-Regel erzeugt werden
	“add System-Parameter”	Ein neuer Parameter soll in ein System-Template integriert werden
	“add System-Import”	Eine neue Import-Quelle soll in ein System-Template integriert werden
JavaScript- und CSS-Editor	Externen Editor aufrufen	Die Datei soll in einer externen Applikation geöffnet werden
	Datei umbenennen	Die Datei soll umbenannt werden
Projekt speichern	Speichern	Im bekannten Pfad speichern. Erzeugte Dateien werden auf Richtigkeit geprüft
	Speichern als	Als neues Projekt speichern. Erzeugte Dateien werden auf Richtigkeit geprüft

Tabelle 5-1: Testfälle für Funktionstest
Quelle: eigene Darstellung

Es wird eine Liste mit den auftretenden Fehlern erstellt. Die Fehler werden nach dem Test behoben. Anschließend werden die betroffenen Funktionen erneut getestet. Bei weitreichenden Änderungen, die Einfluss auf mehrere Funktionen haben könnten, wird die gesamte Testreihe wiederholt.
Während der gesamten Entwicklung des Tools werden Tests durchgeführt, um die ordnungsgemäße Funktion der implementierten Klassen und Methoden zu prüfen. Zum Testen der Funktion wird manuell eine Projektdatei erstellt. Die Ausgabe der Daten auf der Command-Shell zeigt, ob eine Funktion ordnungsgemäß arbeitet.

Praxistest
Nach der Testreihe wird ein Praxistest durchgeführt. Ziel des Praxistests ist die Erstellung eines Extranet-Projektes und ein anschließendes Portieren dieses Projektes in ein Internet-Projekt mit Hilfe des Tools. Die entstandenen Templates werden ins CMS übertragen und werden ab Ende Oktober nach dem Relaunch der Webseite www.gefanuc-europe.com eingesetzt.

Die für das Extranet benötigten System-Templates werden aus den System-Templates fürs Intranet erstellt. Diese werden in das Projekt importiert und mit Hilfe des Template-Editors modifiziert. Für die Start-Templates werden vier Parameter benötigt, die im Tool neu erstellt werden. Die Standard-Templates, sowie einige Spezial-Templates werden neu erstellt. Das Spezial-Template zum Aufbau einer SiteMap wird importiert (Intranet-Template) und leicht modifiziert. CSS und JavaScript werden vom Intranet übernommen. Sie werden ins Projekt übernommen, im entsprechenden Editor umbenannt und durch die festgelegten externen Applikationen bearbeitet. Beim Übertragen der Templates in das CMS ergibt sich

durch das implementierte Fenster zur Anzeige des Template-Codes ein vorher nicht bedachter Vorteil. Der Code kann direkt aus dem Fenster kopiert in das Template-Code-Fenster des Admin-Tools eingefügt werden.
Das Portieren des Extranet-Projektes in das Internet-Projekt wird über die Funktion „save project as“ realisiert, indem das Projekt unter einem anderen Namen in ein neues Verzeichnis gespeichert wird. Nach wenigen Modifizierungen am Import-Template und an CSS und JavaScript sind die Internet-Templates einsatzbereit.

6 Ergebnis und Ausblick

Durch die Überarbeitung der Templates wurde die Verwaltung der Templates wesentlich vereinfacht. Die Realisierung neuer Funktionen konnte mit wesentlich geringerem Aufwand durchgeführt werden, als dies vor der Überarbeitung möglich gewesen wäre. Im Vorfeld wurde seitens der CMS-Administratoren ein solch hohes Maß der Modularisierung nicht erwartet.
Die neuen Funktionen und das überarbeitete Layout förderten zusätzlich die Akzeptanz des CMS innerhalb des Unternehmens.

Das entwickelte Tool TεToo dient zum Verwalten der Templates. Das „Zerlegen“ der Templates in die einzelnen Template-Regeln erleichtert eine Änderung des Codes, besonders bei umfangreichen Templates (z.B. Import-Template), da jede Regel einzeln ausgewählt werden kann. Das Tool wurde speziell auf das im Rahmen der Arbeit entwickelten Template-Konzeptes und das verwendete CMS abgestimmt.
Eine Verwendungsmöglichkeit im Zusammenhang mit anderen auf XML/XSL basierenden Content Management Systemen sollte ohne Änderung des Tool durchaus möglich sein. Mit einigen Abstrichen kann das Tool auch zur Verwaltung von beliebigen XSL-Templates (z.B. die alten Intranet-Templates) verwendet werden.

Das Tool wurde durch das Erstellen neuer Template-Sets durch das Internet- und Extranet-Projekt bereits erfolgreich eingesetzt. Auch die Intranet-Templates wurden in ein Projekt integriert.

Es gibt einige Ideen zur Erweiterung des Tools.

- automatisierte Übertragung der Templates in die CMS-Datenbank.
- Integration weiterer zu einem Web gehörender Dateien (z.B. Java Server Pages, HTML-Seiten).
- Highlighting[31] des XSL-Codes in den entsprechenden Fenstern der Editoren.
- Automatische Einbindung von CSS und JavaScripten in den Template-Code.

Da IC Content in Java realisiert ist wäre sogar eine komplette Integration von TεToo in das Admin-Tool denkbar.

Der ausgiebige Test brachte nicht nur Ideen zur Erweiterung des Tools hervor, sondern deckte auch eine Fehlentscheidung bei der Konzeptionierung auf. Im Nachhinein betrachtet scheint die Speicherung der Projektdaten als Java-Objekt (Klasse Project) sinnvoller, als die realisierte Lösung mit Projektdatei und Templates, da ein Template nur noch selten mit einem anderen Editor geöffnet wird. Die dann benötigte Export-Funktion existiert bereits.

GE Fanuc Automation Europe S.A. plant mittelfristig die Umstellung weiterer Webs von statischen Seiten auf das CMS. Auf diese Umstellung werden die Ergebnisse dieser Arbeit starken Einfluss haben. Das erstellte Template-Konzept ist leicht auf andere Webs zu übertragen. TεToo kann für die Erstellung geeigneter Template-Sets verwendet werden.

[31] Hervorhebung (meist farblich) verschiedener Code-Fragmente bezogen auf ihre Funktion. Ziel ist eine übersichtlichere Quellcode-Darstellung.

7 Anhang

7.1 Übersicht über existierende Templates als Ergebnis der Template-Analyse

Nr.	Template-Name	Informationen	Kategorie	verwendet
1	Icon	Standard-Template für die Navigationsebene "icon"	Start	ja
2	Icon ExLink	dto. für einbinden von externem Inhalt (Frames)	Start-Frame	ja
3	Icon ExLink PopUp	dto. für Anzeigen von externem Inhalt in PopUp-Fenster	Start	ja
4	News	für Abteilungsstartseiten; zeigt Inhalte in speziellem Format (News) an	Start	ja
5	Register	Standard-Template für die Navigationsebene "register"	Start	ja
6	Register ExLink	dto. für einbinden von externem Inhalt (Frames)	Start-Frame	ja
7	Register ExLink PopUp	dto. für Anzeigen von externem Inhalt in PopUp-Fenster	Start	ja
8	SiteMap	erzeugt eine Seitenübersicht	Spezial	ja
9	Standard	Standard-Template für die oberen Navigationsebenen	Start	ja
10	Standard ExLink	dto. für einbinden von externem Inhalt (Frames)	Start-Frame	ja
11	Standard ExLink PopUp	dto. für Anzeigen von externem Inhalt in PopUp-Fenster	Start	ja
12	Standard Frame	erstellt Frameset für Spezialseiten (GE-Link)	Start-Frame	ja
13	Switch	Standard-Template für die Navigationsebene "switch"	Start	ja
14	Switch ExLink	dto. für einbinden von externem Inhalt (Frames)	Start-Frame	ja
15	Switch ExLink PopUp	dto. für Anzeigen von externem Inhalt in PopUp-Fenster	Start	ja
16	System_Import	enthält die QuickSearch-Funktionalität	Spezial	ja
17	Attribute with header	erzeugt eine Tabelle aus Content-Attributen	Start	ja
18	BLUE Icon Nav. With Header	nur zur Demonstration		nein
19	BLUE Reg. Nav. With Header	nur zur Demonstration		nein
20	BLUE Standard with Header	nur zur Demonstration		nein
21	BLUE Switch to Reg. Nav. With Header	nur zur Demonstration		nein
22	BLUE webleiste	nur zur Demonstration		nein
23	DigitalCockpit Reg. Nav. W. Header	erzeugt spezielle Tabellen für die Digital-Cockpit-Anwendung	Start-Spezial	ja
24	Icon Frameset Exlink Danny	nur für Testzwecke		nein

25	Icon New Danny	nur für Testzwecke		nein
26	Import	enthält Regeln für die Fußnote und das Hauptmenü	Spezial	ja
27	Menu	enthält Regeln für die linke Navigationsleiste und die Content-Darstellung		nein
28	Navigation To Navigation	springt vom aktuellen auf einen anderen Navigationspunkt	Start-Spezial	ja
29	Plain Content Without Navigation	Content-Darstellung ohne Aufbau von Menüs	Spezial	ja
30	Project Overview e-Team	erzeugt Tabelle aus Content-Attributen für das eBusiness-Team	Start-Spezial	ja
31	Register Frameset Exlink Danny	nur für Testzwecke		nein
32	Register New Danny	nur für Testzwecke		nein
33	Register Table Attributes	erzeugt eine Tabelle aus Content-Attributen	Start	ja
34	search	Klon des "alten" Intranet mit Suchfunktion		nein
35	Standard Attribute	zeigt die Attribute eines Contents an		nein
36	Standard Frame Image	enthält Regeln für das Menübild bei Frame-Darstellung		nein
37	Standard Frame Left	enthält Regeln für die linke Navigationsleiste bei Frame-Darstellung		nein
38	Standard Frame Top	enthält Regeln für den oberen Bereich bei Frame-Darstellung		nein
39	Standard Frameset Exlink Danny	nur für Testzwecke		nein
40	Standard NEW Danny	nur für Testzwecke		nein
41	Standard New Frame Content Danny	nur für Testzwecke		nein
42	Standard NEW Frame Danny	nur für Testzwecke		nein
43	Standard Show Media	zeigt die Attribute und Medien eines Contents an		nein
44	Sub Icon Menue	bietet eine Navigationsebene unter der "icon"-Ebene		nein
45	Switch Frameset Danny Exlink	nur für Testzwecke		nein
46	Switch New Danny	nur für Testzwecke		nein
47	webleiste	??		nein
48	Project tables	erzeugt Tabelle aus Content-Attributen für die Abteilung AEB&FIT	Start-Spezial	ja
49	Automative Project Downloads	zeigt eine Download-Auswahl verschiedener Contents (Medien) an		nein
50	Icon Frameset CMS Content	erzeugt Frameset zur Darstellung von CMS-basiertem Content in der Navigationsebene "icon"	Start-Frame	ja
51	Register Frameset CMS Content	erzeugt Frameset zur Darstellung von CMS-basiertem Content in der Navigationsebene "register"	Start-Frame	ja
52	Standard Frameset CMS Content	erzeugt Frameset zur Darstellung von CMS-basiertem Content in den	Start-Frame	ja

		oberen Navigationsebenen		
53	Switch Frameset CMS Content	erzeugt Frameset zur Darstellung von CMS-basiertem Content in der Navigationsebene "switch"	Start-Frame	ja
54	Quality Icon ISO 9000	wie "Icon", Icon-Menüeinträge sind untereinander angeordnet	Start	ja
55	Quality Register ISO 9000	wie "Register", Icon-Menüeinträge sind untereinander angeordnet	Start	ja
56	Ampliefier table webdisplay	erzeugt Tabelle aus Content-Attributen für das TechCenter		nein
57	Icon Frame Left	enthält Regeln für die linke Navigationsleiste bei Frame-Darstellung in der "icon"-Ebene	Frame	ja
58	Icon Frame Upper Register	enthält Regeln für den Register-Bereich bei Frame-Darstellung in der "icon"-Ebene	Frame	ja
59	Register Frame Left	enthält Regeln für die linke Navigationsleiste bei Frame-Darstellung in der "register"-Ebene	Frame	ja
60	Register Frame Upper Register	enthält Regeln für den Register-Bereich bei Frame-Darstellung in der "register"-Ebene	Frame	ja
61	Standard Frame Department Navigation	enthält Regeln für den oberen Bereich bei Frame-Darstellung	Frame	ja
62	Standard Frame Left	enthält Regeln für den Register-Bereich bei Frame-Darstellung in den oberen Ebenen	Frame	ja
63	Switch Frame Left	enthält Regeln für die linke Navigationsleiste bei Frame-Darstellung in der "switch"-Ebene	Frame	ja
64	Switch Frame Upper Register	enthält Regeln für den Register-Bereich bei Frame-Darstellung in der "switch"-Ebene	Frame	ja
65	New Standard	nur für Testzwecke		nein
67	Icon without header	wie "Icon", in der Content-Darstellung wird keine Headline angezeigt	Start	ja
68	Register without header	wie "Register", in der Content-Darstellung wird keine Headline angezeigt	Start	ja
69	Standard without header	wie "Standard", in der Content-Darstellung wird keine Headline angezeigt	Start	ja
70	Switch without header	wie "Switch", in der Content-Darstellung wird keine Headline angezeigt	Start	ja
71	Show external Graphic in Reg Nav	wie "Register", zeigt statt Content eine externe Grafik	Start	ja
72	Show external Graphic in Switch Nav	wie "Switch", zeigt statt Content eine externe Grafik	Start	ja
73	Show external Graphic in Icon Nav	wie "Icon", zeigt statt Content eine externe Grafik	Start	ja

Tabelle 7-1: Übersicht über existierende Templates als Ergebnis der Templateanalyse
Quelle: eigene Darstellung

7.2 Template-Übersicht als Ergebnis der Überarbeitung

Standard-Templates

Nr.	Template-Name	Used for
1	icon	Icon menu item; standard content view; horizontal icon-adjustment
2	icon-verIcon	Icon menu item; standard-content view; vertical icon-adjustment
3	register	register menu item; standard content view; horizontal icon-adjustment
4	register-attributetable	register menu item; content's attribute table view; horizontal icon-adjustment
5	register-verIcon	register menu item; standard content view; vertical icon-adjustment
6	standard	standard menu item; standard content view
7	standard-news	standard menu item; news-content view
8	standard-SiteMap	shows navigation structure of chosen department's site
9	switch	switch menu item; standard content view

Tabelle 7-2: überarbeitete Standard-Templates
Quelle: eigene Darstellung

Spezial-Templates

Nr.	Template-Name	Used for
1	register-Digital Cockpit Table	Shows specific sheets for staff
2	register-Automotive-Projectwizard Conversion Tracking	shows content's attributes as table and offers a link to projecttool
3	register-AEB&FIT-Projectwizard	shows content's attributes as table and offers a link to projecttool
4	register-IT-eTeam-Projectwizard	shows content's attributes as table and offers a link to projecttool

Tabelle 7-3: überarbeitete Spezial-Templates
Quelle: eigene Darstellung

System-Templates

Nr.	Template-Name	Used for
1	printView	creates printerfriendly view of actual page's contents
2	Plain Content Without Navigation	shows contents without navigation menus
3	import	import-Template used by all other templates

Tabelle 7-4: überarbeitete System-Templates
Quelle: eigene Darstellung

7.3 Quellcode eines Standard-Templates (standard-attributetable)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--name: standard-attributetable-->
<!--web: _intra-->
<!--version: 0.00-->
<!--changed: Mon Nov 10 16:43:54 CET 2003-->
<!--description: null-->
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:java="http://xml.apache.org/xslt/java"
xmlns:xalan="http://xml.apache.org/xalan" exclude-result-
prefixes="xalan">

<!--<xsl:import>import.xsl</xsl:import>-->
<xsl:import href="ictdb://3de70d6b-2f7f-b1fa-168a-
018cc0298603#d888b1e0-7336-497b-a571-c36257dcccc3#7ff712f9-3e4c-
38e1-1127-2164c01e0811"/>

<xsl:output method="html" version="4.0" encoding="ISO-8859-1"
indent="yes"/>

  <xsl:param name="Type">register</xsl:param>
  <!--to control the appearence of the current menu item in the
navigation bar structure-->

  <xsl:param name="Content">attributetable</xsl:param>
  <!--to control how the contents connected to this menu item are
displayed-->

  <xsl:param name="IconNav"/>
  <!--to control the adjustment of the menu items in the icon
navigation bar-->

  <xsl:param name="AttrSort">Project
Name,LED,Status,Responsibility,Priority,Delivery date,Planned
date,Remark,</xsl:param>
  <!--How to sort contents in an attributetable. Comma seperated
list, has to end with a Comma-->

  <xsl:param
name="ProjectToolLink">http://webadm.gefanuceur.ge.com/project/?uMen
=</xsl:param>
  <!--Link to the projecttool linked to an attributetable-->

  <xsl:template name="start" match="/">
  <!--Standard-Template-Rule-->
    <xsl:call-template name="master"/>
  </xsl:template>


  <xsl:template name="" match="/">
  <!--no description-->
    <xsl:call-template name="master"/>
  </xsl:template>
</xsl:stylesheet>
```

7.4 CMS-Dokumentation für CMS-Redakteure und Benutzer

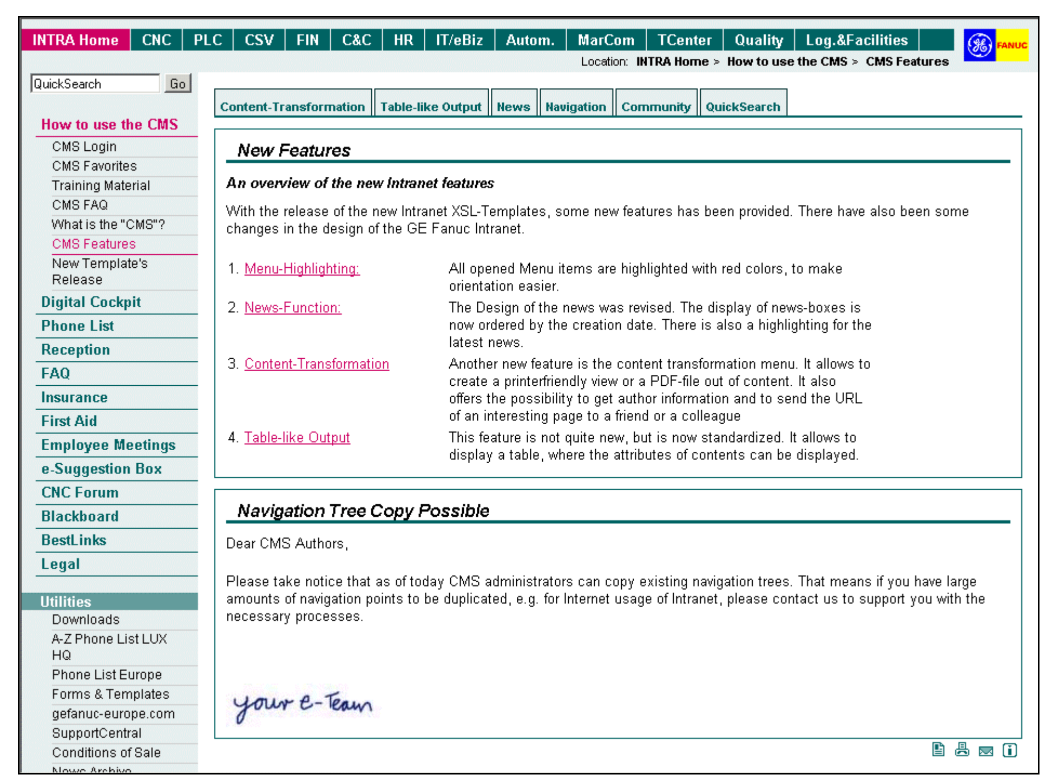


Abbildung 7-1: CMS-Dokumentation für Benutzer 1
Quelle: GE Fanuc Intranet

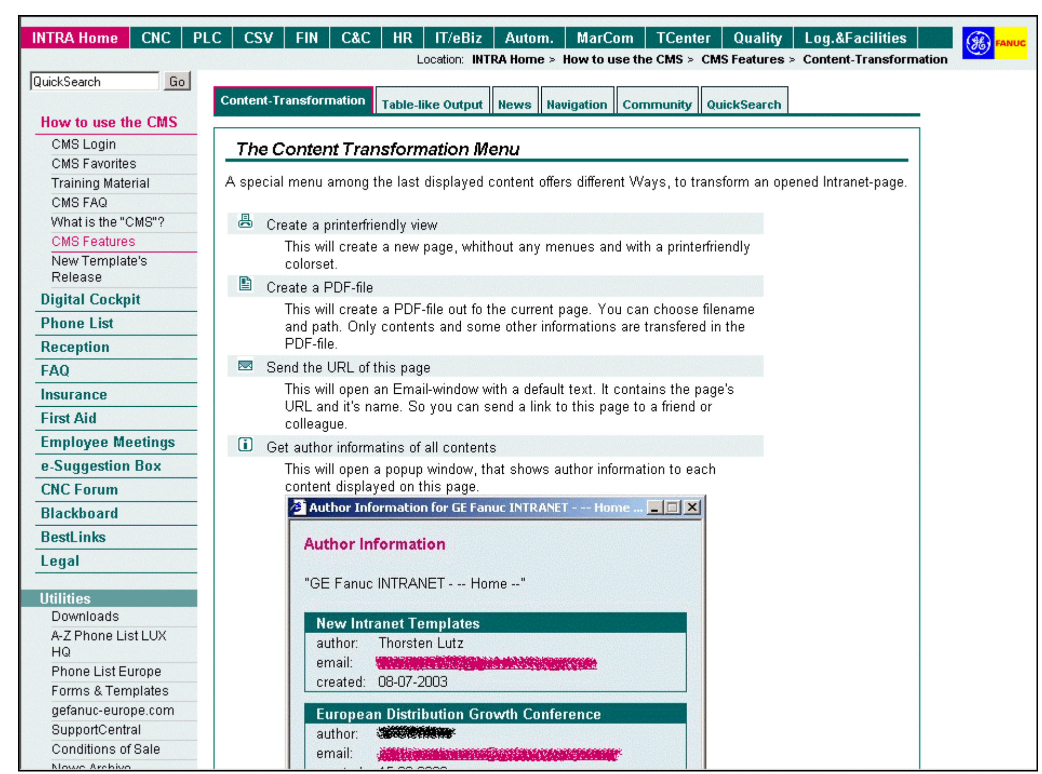


Abbildung 7-2: CMS-Dokumentation für Benutzer 2
Quelle: GE Fanuc Intranet

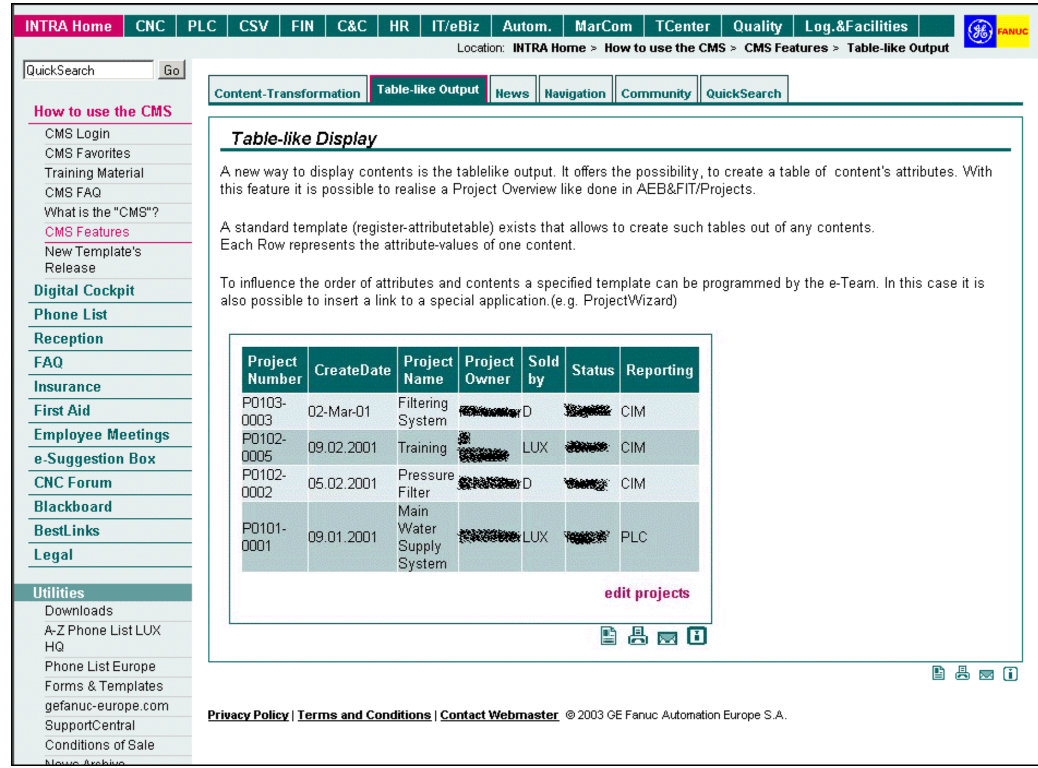


Abbildung 7-3: CMS-Dokumentation für Benutzer 3
Quelle: GE Fanuc Intranet

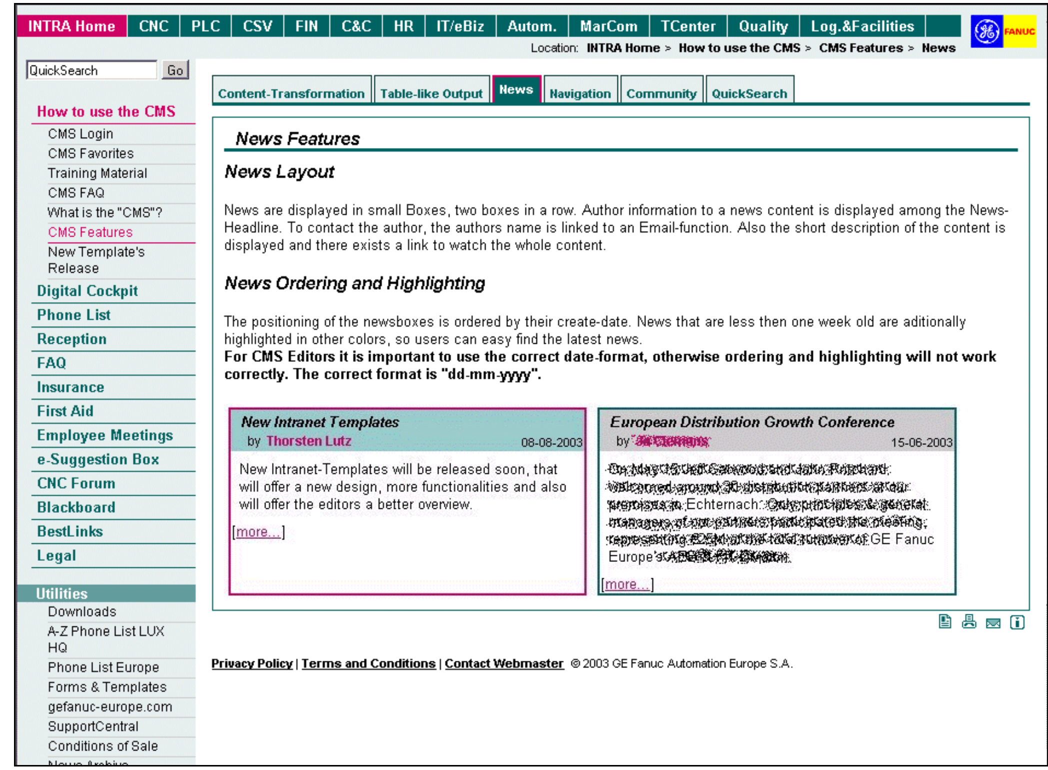


Abbildung 7-4: CMS-Dokumentation für Benutzer 4
Quelle: GE Fanuc Intranet

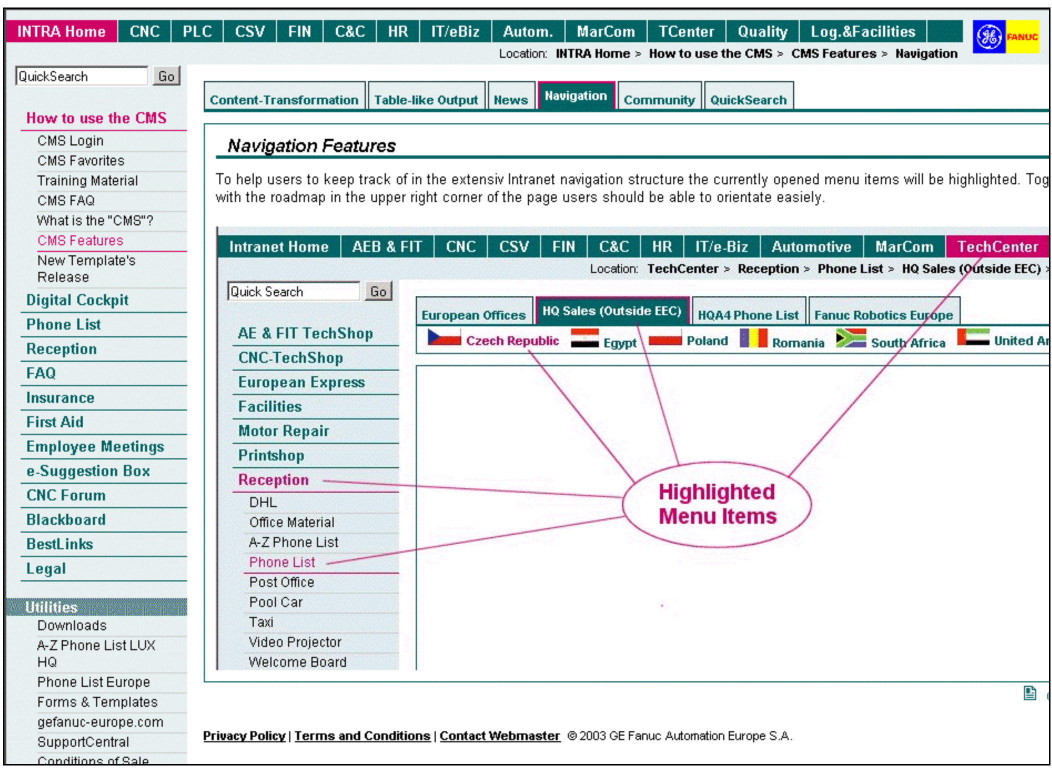


Abbildung 7-5: CMS-Dokumentation für Benutzer 5
Quelle: GE Fanuc Intranet

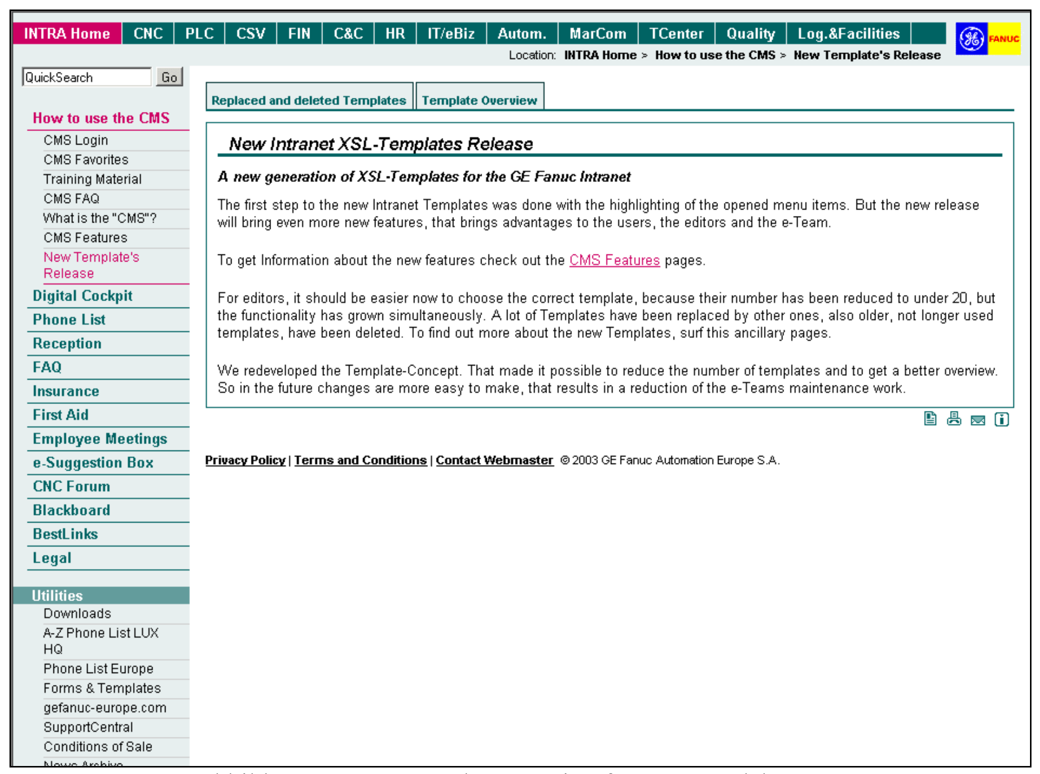
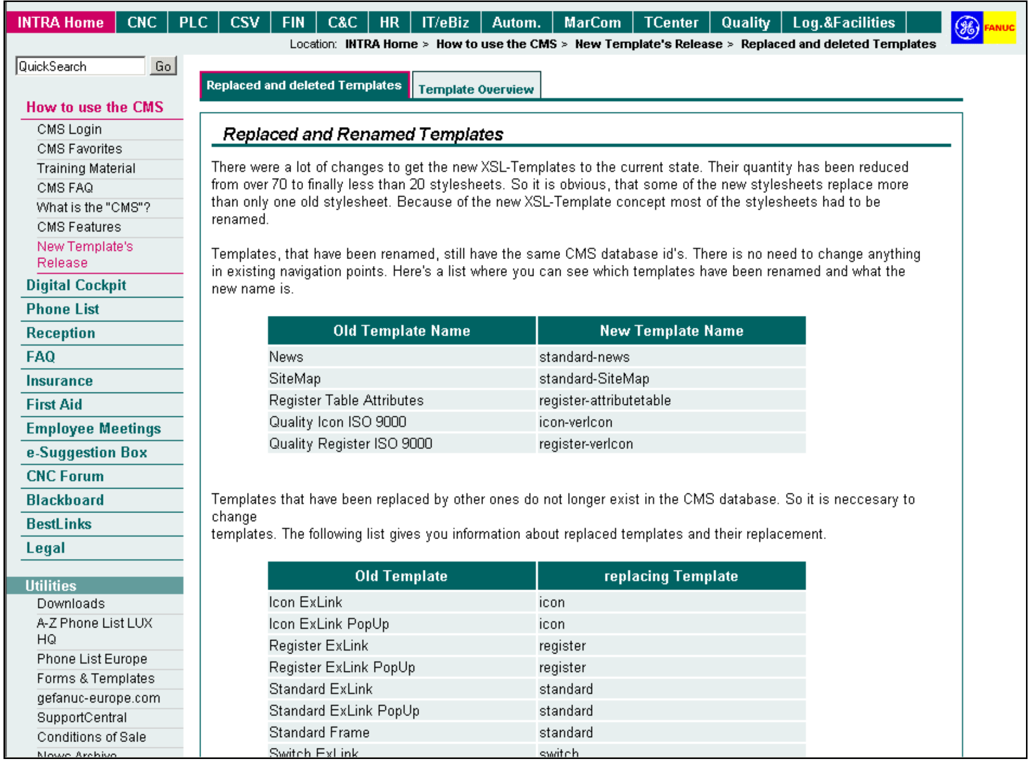


Abbildung 7-6: CMS-Dokumentation für CMS-Redakteure 1
Quelle: GE Fanuc Intranet



INTRA Home | CNC | PLC | CSV | FIN | C&C | HR | IT/eBiz | Autom. | MarCom | TCenter | Quality | Log.&Facilities

Location: INTRA Home > How to use the CMS > New Template's Release > Replaced and deleted Templates

QuickSearch Go

How to use the CMS
- CMS Login
- CMS Favorites
- Training Material
- CMS FAQ
- What is the "CMS"?
- CMS Features
- New Template's Release

Digital Cockpit
Phone List
Reception
FAQ
Insurance
First Aid
Employee Meetings
e-Suggestion Box
CNC Forum
Blackboard
BestLinks
Legal

Utilities
- Downloads
- A-Z Phone List LUX HQ
- Phone List Europe
- Forms & Templates
- gefanuc-europe.com
- SupportCentral
- Conditions of Sale

Replaced and deleted Templates | Template Overview

Replaced and Renamed Templates

There were a lot of changes to get the new XSL-Templates to the current state. Their quantity has been reduced from over 70 to finally less than 20 stylesheets. So it is obvious, that some of the new stylesheets replace more than only one old stylesheet. Because of the new XSL-Template concept most of the stylesheets had to be renamed.

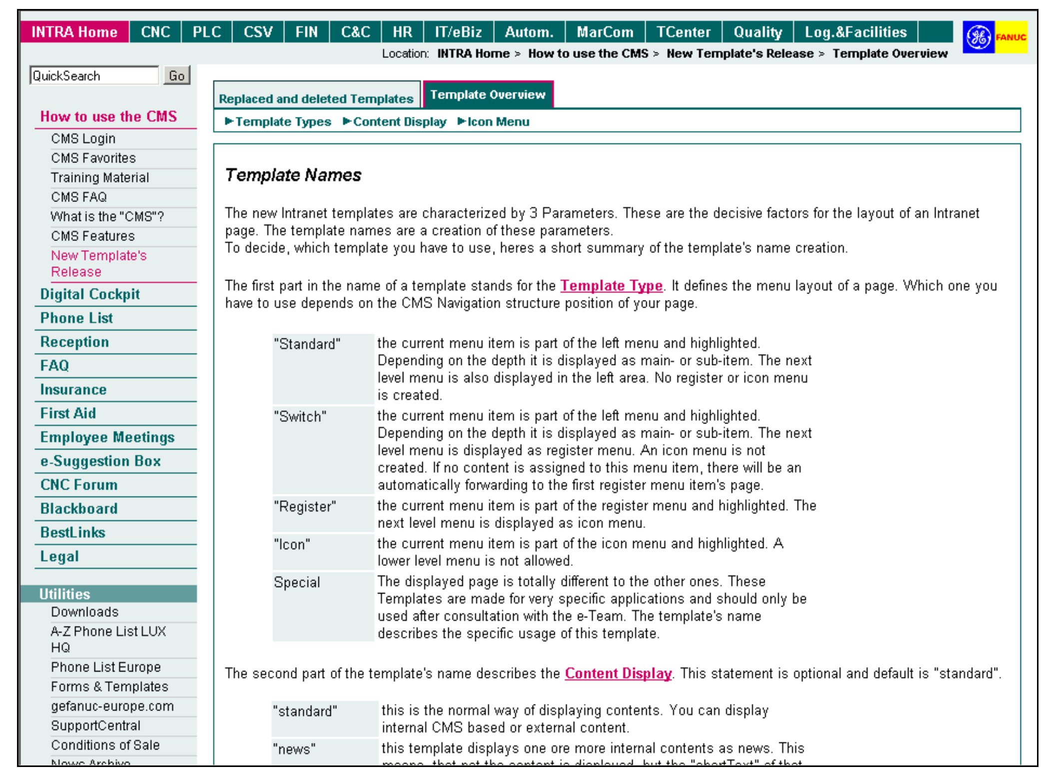
Templates, that have been renamed, still have the same CMS database id's. There is no need to change anything in existing navigation points. Here's a list where you can see which templates have been renamed and what the new name is.

Old Template Name	New Template Name
News	standard-news
SiteMap	standard-SiteMap
Register Table Attributes	register-attributetable
Quality Icon ISO 9000	icon-verIcon
Quality Register ISO 9000	register-verIcon

Templates that have been replaced by other ones do not longer exist in the CMS database. So it is neccesary to change
templates. The following list gives you information about replaced templates and their replacement.

Old Template	replacing Template
Icon ExLink	icon
Icon ExLink PopUp	icon
Register ExLink	register
Register ExLink PopUp	register
Standard ExLink	standard
Standard ExLink PopUp	standard
Standard Frame	standard
Switch ExLink	switch

Abbildung 7-7: CMS-Dokumentation für CMS-Redakteure 2
Quelle: GE Fanuc Intranet



INTRA Home | CNC | PLC | CSV | FIN | C&C | HR | IT/eBiz | Autom. | MarCom | TCenter | Quality | Log.&Facilities

Location: INTRA Home > How to use the CMS > New Template's Release > Template Overview

QuickSearch Go

How to use the CMS
- CMS Login
- CMS Favorites
- Training Material
- CMS FAQ
- What is the "CMS"?
- CMS Features
- New Template's Release

Digital Cockpit
Phone List
Reception
FAQ
Insurance
First Aid
Employee Meetings
e-Suggestion Box
CNC Forum
Blackboard
BestLinks
Legal

Utilities
- Downloads
- A-Z Phone List LUX HQ
- Phone List Europe
- Forms & Templates
- gefanuc-europe.com
- SupportCentral
- Conditions of Sale

Replaced and deleted Templates | Template Overview

▸Template Types ▸Content Display ▸Icon Menu

Template Names

The new Intranet templates are characterized by 3 Parameters. These are the decisive factors for the layout of an Intranet page. The template names are a creation of these parameters.
To decide, which template you have to use, heres a short summary of the template's name creation.

The first part in the name of a template stands for the Template Type. It defines the menu layout of a page. Which one you have to use depends on the CMS Navigation structure position of your page.

"Standard"	the current menu item is part of the left menu and highlighted. Depending on the depth it is displayed as main- or sub-item. The next level menu is also displayed in the left area. No register or icon menu is created.
"Switch"	the current menu item is part of the left menu and highlighted. Depending on the depth it is displayed as main- or sub-item. The next level menu is displayed as register menu. An icon menu is not created. If no content is assigned to this menu item, there will be an automatically forwarding to the first register menu item's page.
"Register"	the current menu item is part of the register menu and highlighted. The next level menu is displayed as icon menu.
"Icon"	the current menu item is part of the icon menu and highlighted. A lower level menu is not allowed.
Special	The displayed page is totally different to the other ones. These Templates are made for very specific applications and should only be used after consultation with the e-Team. The template's name describes the specific usage of this template.

The second part of the template's name describes the Content Display. This statement is optional and default is "standard".

"standard"	this is the normal way of displaying contents. You can display internal CMS based or external content.
"news"	this template displays one ore more internal contents as news. This

Abbildung 7-8: CMS-Dokumentation für CMS-Redakteure 3
Quelle: GE Fanuc Intranet

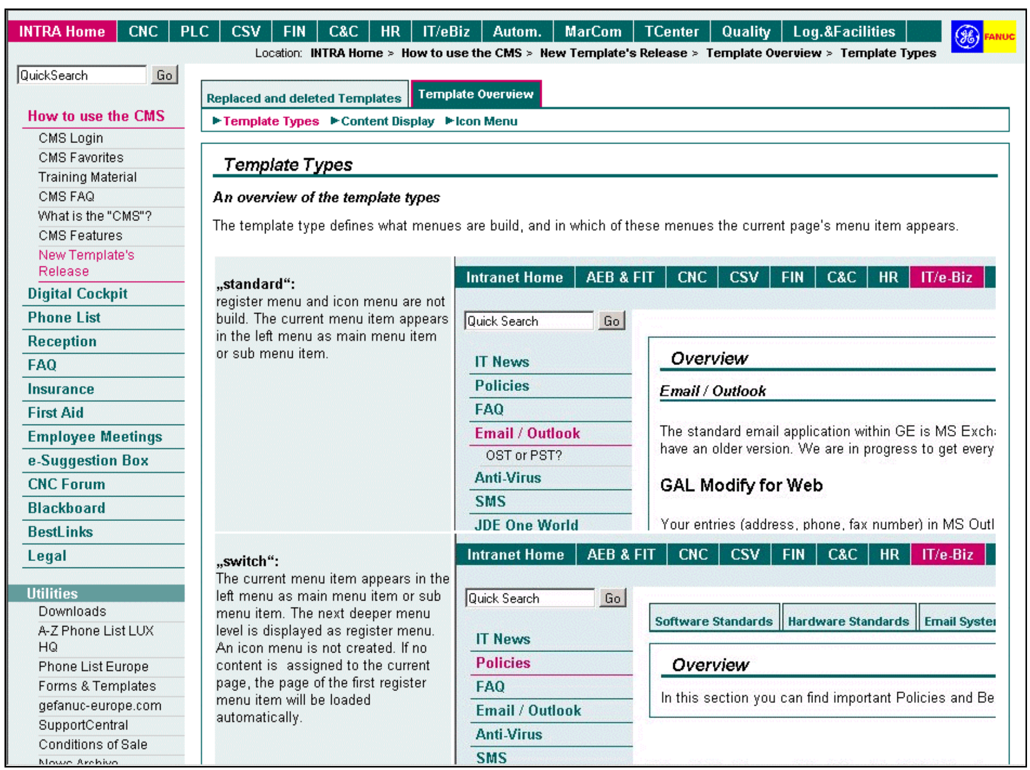


Abbildung 7-9: CMS-Dokumentation für CMS-Redakteure 4
Quelle: GE Fanuc Intranet

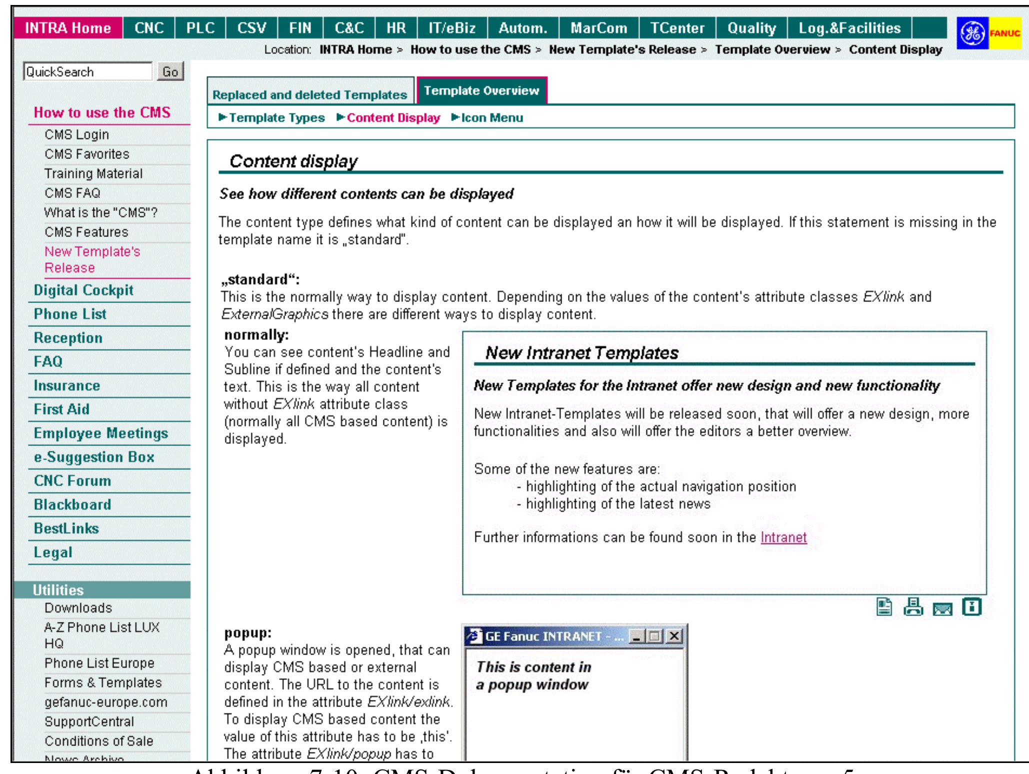


Abbildung 7-10: CMS-Dokumentation für CMS-Redakteure 5
Quelle: GE Fanuc Intranet

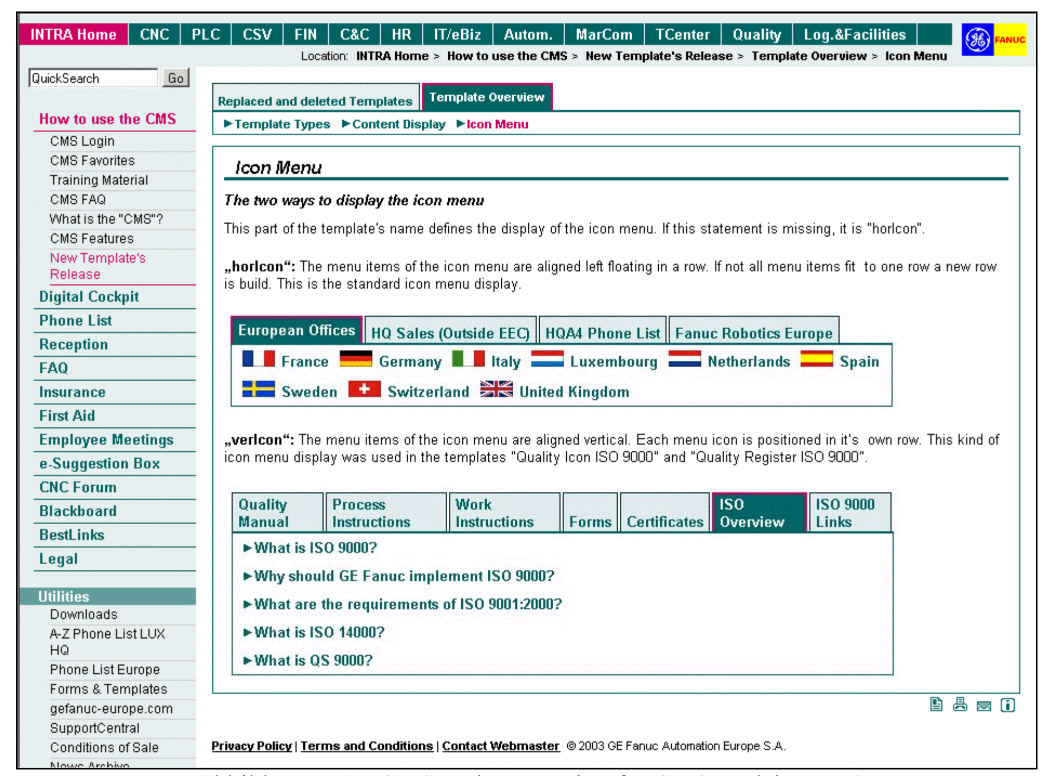


Abbildung 7-11: CMS-Dokumentation für CMS-Redakteure 6
Quelle: GE Fanuc Intranet

7.5 Ablaufdiagramm der Template-Abarbeitung

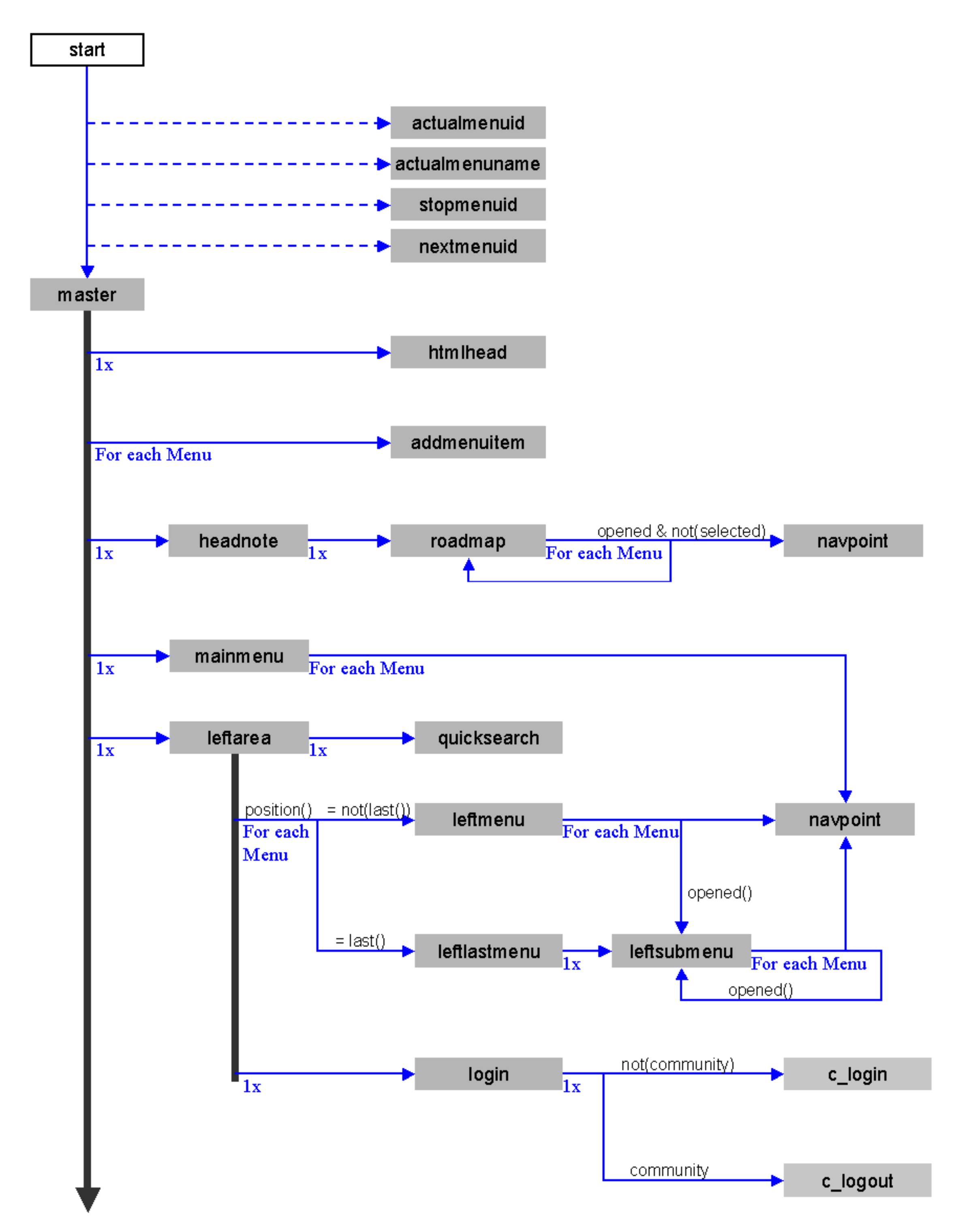


Abbildung 7-12: Ablaufdiagramm: Template-Abarbeitung (Intranet) 1
Quelle: eigene Darstellung

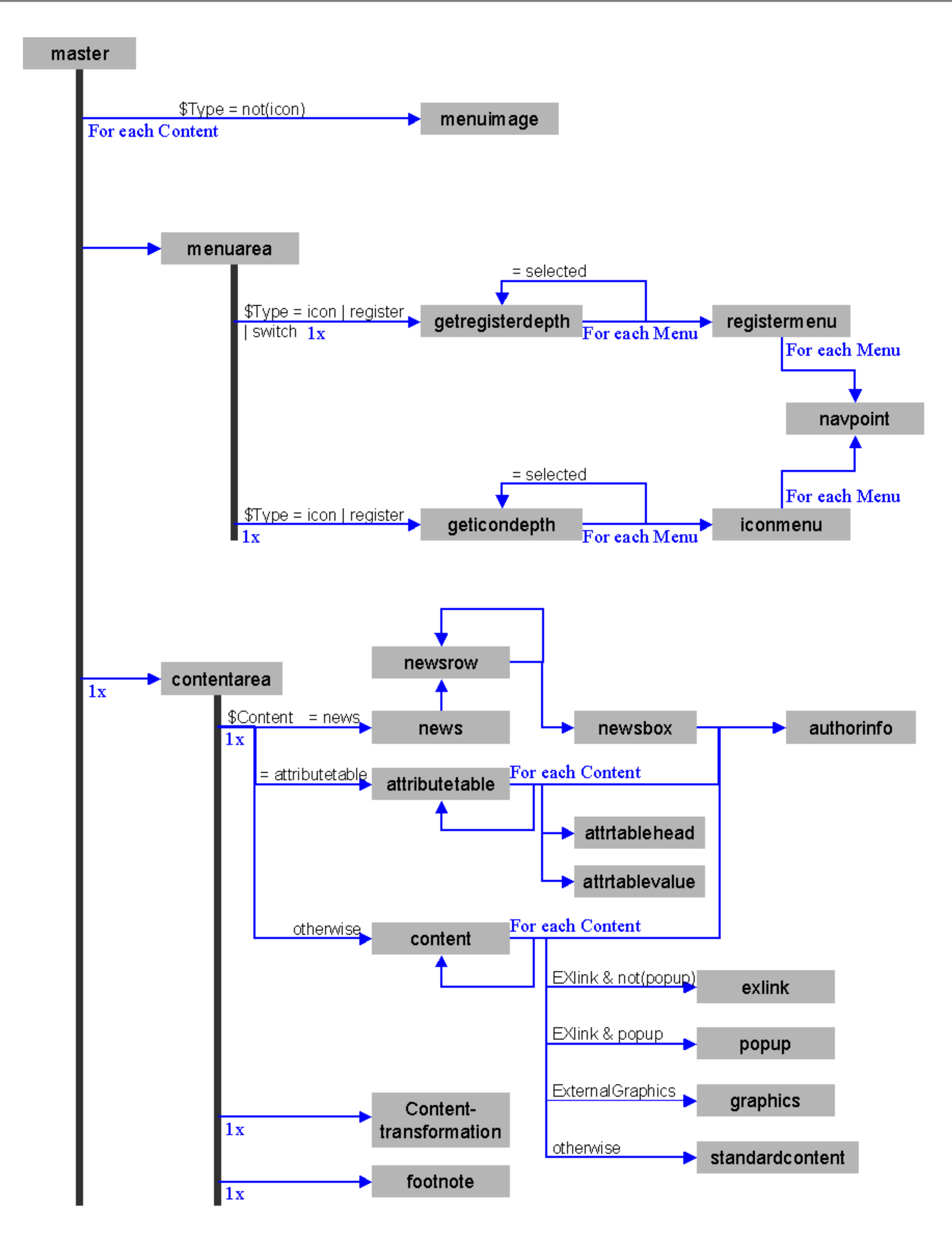


Abbildung 7-13: Ablaufdiagramm: Template-Abarbeitung (Intranet) 2
Quelle: eigene Darstellung

7.6 TεToo Benutzerhandbuch

TεToo 1.0

Template Tool
von Thorsten Lutz



Benutzerhandbuch





FB Angewandte Informatik
Fachhochschule Trier, Standort Schneidershof, D-54293 Trier

Copyright

Copyright © 2003 GE Fanuc Automation Europe S.A.

Haftungsausschluss

Hiermit wird jede Haftung ausgeschlossen, die Schäden betreffen, welche durch den Einsatz dieser Software hervorgerufen werden, insbesondere Schäden an Dateien, die von der Software bearbeitet wurden.

Sicherheitshinweise

1. Lesen Sie dieses Benutzerhandbuch, bevor Sie die Software verwenden.

2. Die Software wurde für das Betriebssystem Microsoft Windows® entwickelt. Eine ordnungsgemäße Funktion unter anderen Betriebssystemen (z.B. Linux, Solaris,...) kann nicht gewährleistet werden.

- 2 -

Inhaltsverzeichnis

1. Vorwort

Dieses Tool wurde im Rahmen meiner Diplomarbeit „Entwicklung eines Tools zur Verwaltung und Generierung von XSL-Templates“ bei GE Fanuc Automation S.A. in Echternach entwickelt.
Die Diplomarbeit hat das Ziel, ein sinnvolles Konzept zur Nutzung von XSL-Templates für die GE Fanuc Webauftritte, basierend auf einem Content Management System (CMS), zu entwickeln und umzusetzen. Dieses Tool wurde entwickelt, um die entstandenen Templates zu verwalten.

1.1 Wichtige Hinweise

1. Vor der Nutzung dieses Tools sollten Sie sich mit den Technologien HTML, XML und XSL vertraut machen. Diese bilden die Grundlage zu einem sinnvollen Einsatz des Programms.

2. Lesen Sie vor der Benutzung des Tools dieses Benutzerhandbuch aufmerksam, um Fehler zu vermeiden.

3. Das Tool ist in der Lage, Dateien zu ändern oder zu löschen. Erstellen Sie im Zweifelsfall Sicherungskopien von Dateien und Ordnern, auf die das Programm zugreift. Dies gilt besonders für XSL-Dateien, die vom Programm importiert werden.

4. Dieses Tool ist in Java programmiert, das heißt es sollte auf beliebigen Betriebssystemen lauffähig sein. Es wurde unter Windows® entwickelt und nur dort getestet. Die Lauffähigkeit unter anderen Betriebssystemen (z.B. Linux, Solaris,...) kann daher nicht gewährleistet werden.

2. Zweck des Tools

In größeren Unternehmen werden zum Aufbau von Webseiten zunehmend Content-Management-Systeme (CMS) verwendet, deren großer Vorteil eine Trennung von Inhalt und Layout darstellt. Es ist für einen Benutzer der Inhalte auf einer Webseite präsentieren will nicht zwingend notwendig HTML und andere Web-Sprachen zu beherrschen.

Die Inhalte werden von den CMS-Systemen, meist in XML-Format, in einer Datenbank verwaltet. Um diese XML-Dokumente in HTML-Code umzuwandeln, so dass sie in einem Browser darstellbar sind, werden Schablonen oder Vorlagen benötigt, die das Layout der Webseite bestimmen und die Inhalte an den richtigen Positionen einfügen. Zu diesem Zweck werden XSL-Templates verwendet.

Für die Pflege eines Webauftritts (nicht für die Inhalte), und damit auch für die Verwaltung der XSL-Templates, ist in den meisten Fällen ein kleiner Personenkreis verantwortlich. Diese Personen sind in der Regel Web-Designer oder IT-Fachkräfte, die mit dem eingesetzten CMS-System bestens vertraut sind.
Daher werden Funktionalitäten zur Verwaltung der Templates von vielen CMS-Anbietern vernachlässigt und selten stehen gute Tools zur Verfügung.

Bei Unternehmen, die eine ausgeprägte Web-Struktur benötigen, kann eine große Anzahl von Templates zum Einsatz kommen. Der Aufwand zur Pflege dieser Templates wird damit sehr groß.
Die verwendeten Templates können oftmals in Gruppen (Template-Sets) zusammengefasst werden. Für jedes Web gibt es ein Template-Set (z.B. Intranet, Extranet, Internet).

Dieses Tool erleichtert die Verwaltung von Template-Sets und bietet Funktionen zum Ändern, Erstellen und Entfernen von XSL-Templates in einem Template-Set. Darüber hinaus berücksichtigt das Tool auch andere Dateien, die zum Aufbau einer Webpräsenz benötigt werden (z.B. Cascading-Stylesheets).

3. Struktur eines Template-Sets

Ein Template-Set setzt sich aus mehreren XSL-Templates zusammen. Dabei unterscheidet man drei Arten von Templates. Die System-Templates sind für den Aufbau einer Webseite, deren Layout und Funktionalität verantwortlich. Sie können eigenständig arbeiten, oder von anderen Templates (Start-Templates) ausgeführt werden. In den Start-Templates, die in Standard-Templates und Special-Templates aufgeteilt sind, werden Parameter gesetzt, die für das Layout einer Webseite maßgeblich sind. Die Start-Templates beinhalten eine Standard-Template-Regel, die ein System-Template mit den festgelegten Parametern ausführt. Spezial-Templates können zusätzliche Template-Regeln (Template Rules) beinhalten, die die Funktion der System-Templates ersetzen oder erweitern.
Zusätzlich kann ein Template-Set CSS- und JavaScript-Dateien beinhalten. Die Informationen über ein Template-Set werden in einer Projektdatei gespeichert. Jedes Template-Set liegt idealerweise in einem eigenen Dateiordner.

3.1 Aufbau eines Templates

Die Software erzeugt XSL-Templates, die einem bestimmten Schema entsprechen. Die Hauptteile eines XSL-Templates bilden die Import-Angaben, die Parameter und die Template-Regeln. Die Inhalte der Hauptteile hängen vom Template-Typ ab. Zusätzlich gibt es Tags, die für ein XSL-Template unverzichtbar sind, der <xsl:stylesheet>-Tag, der den Template-Code einschließt, sowie den <xsl-output>-Tag, der angibt, in welchem Format der Ausgabecode erstellt werden soll (HTML). Das Tool fügt zusätzlich Kommentare ein, die Informationen über das Template enthalten. Das folgende Syntaxdiagramm verdeutlicht den Aufbau eines Templates zu.

```
XSL-Template ::=        '<?xml version="1.0" encoding="ISO-8859-1"?>'
                        <INFORMATIONS>
                        '<xsl:stylesheet version="1.0"
                        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                        xmlns:java="htt://xml.apache.org/xslt/java"
                        xmns:xalan="http://xml.apache.org/xalan"
                        exclude-result-prefixes="xalan">'
                        <IMPORTS>
                        '<xsl:output method="html" version="4.0" encoding="ISO-
                        8859-1" indent="yes"/>'
                        <PARAMETERS>
                        <TEMPLATE-RULES>
                        '</xsl:stylesheet>'

INFORMATIONS ::=        '<!--name: ' <TEMPLATENAME> '-->'
                        '<!--web: ' <PROJECTNAME> '-->'
                        '<!--version: ' <VERSION> '-->'
                        '<!--changed: ' <CHANGEDATE> '-->'
                        '<!--description: ' <DESCRIPTION> '-->'

TEMPLATENAME ::=        [CDATA] Name des Templates

PROJECTNAME ::=         [CDATA] Name des Projekts

VERSION ::=             [CDATA] Angabe der Templateversion

CHANGEDATE ::=          [CDATA] Datum der letzten Änderung (z.B. 'Thu Sep 25
                        14:23:35 CEST 2003')

DESCRIPTION ::=         [CDATA] Beschreibung der Templatefunktion
```

- 6 -

```
IMPORTS ::=           { '<xsl:import href="' <IMPORTINFO> '"/>' }

IMPORTINFO ::=        [CDATA] Pfad zur Import-Quelle

PARAMETERS ::=        '<xsl:param name="' <P-NAME> '">' <P-VALUE>
                       '</xsl:param>'
                      '<!--' <P-DESCRIPTION> '-->'

P-NAME ::=            [CDATA] Name des Parameters

P-VALUE ::=           [CDATA] Wert des Parameters | { <XSL-TAG> }

P-DESCRIPTION ::=     [CDATA] Beschreibung der Parameter-Funktion

TEMPLATE-RULES ::=    '<xsl:template name="' <RULE-NAME> '" match="'
                      <RULE-MATCH> '">'
                      '<!--' <RULE-DESCR> '-->'
                      { <XSL-TAG> }
                      '</xsl:template>'

RULE-NAME ::=         [CDATA] Name der Template-Regel

RULE-MATCH ::=        <XPATH-COMMAND> | ''

RULE-DESCR ::=        [CDATA] Beschreibung der Template-Regel

XSL-TAG ::=           XSL-Befehl, evtl. mit Kindknoten

XPATH-COMMAND ::=     XPath-Befehl zum Selektieren von XML-Knoten
```

Das Tool bietet die Möglichkeit, XSL-Templates zu importieren. Dabei werden alle Knoten des XSL-Dokuments, die nicht in das obige Syntaxdiagramm integriert werden können, ignoriert.

3.2 Die Projektdatei

In der Projektdatei werden alle Informationen über ein Projekt gespeichert. Bei jedem Speichern des Projektes wird diese Projektdatei in XML-Form neu erzeugt. Die Projektdatei trägt den Namen des Projekts mit der Dateiendung „.ttp“.
Hier ein Beispiel für eine Projektdatei:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Project name="inter_UK">
       <ProjectDescription>Template set for the GE Fanuc Automation UK web
       </ProjectDescription>
       <StandardTRuleName name="start"/>
<LastChange date="Thu Sep 25 14:23:35 CEST 2003"/>

       <Import source="/systemstatic/_INTER/import.xsl" commented="yes"/>
       <Import source="ictdb://3de70d6b-2f7f-b1fa-168a-018cc0298603"
       commented="no"/>

       <StandardParameter name="Type" comment="to control the appearence of
       the current menu item in the navigation bar structure">
              <PossibleValue value="standard"/>
              <PossibleValue value="switch"/>
              <PossibleValue value="register"/>
              <PossibleValue value="icon"/>
       </StandardParameter>
       <StandardParameter name="RightBox" comment="to decide if a news box on
       the right side has to be displayed">
              <PossibleValue value="yes"/>
              <PossibleValue value="no"/>
              <PossibleValue value="*no value*"/>
       </StandardParameter>
       <SpecialParameter name="Country" comment="to control country specific
       display options" value="uk">
              <PossibleValue value="*any value*"/>
       </SpecialParameter>

       <StandardTemplate name="standard.xsl" path="."/>
       <StandardTemplate name="standard-news.xsl" path="."/>
       <StandardTemplate name="news-archive.xsl" path="."/>
       <StandardTemplate name="register.xsl" path="."/>
       <SpecialTemplate name="standard-SiteMapFull.xsl" path="."/>
       <SystemTemplate name="import.xsl" path="."/>
       <SystemTemplate name="printView.xsl" path="."/>

       <CSSFile name="INTERnet_styles.css" path="."/>
       <JSFile name="INTERnet_scripts.js" path="."/>
</Project>
```

4. Installation des Programms

Das Tool liegt im Normalfall als JAR-Archiv vor.

Um das Tool nutzen zu können, muss auf Ihrem Rechner das „Java Runtime Environment" (JRE) in der Version 1.4 oder höher installiert sein.
Unter http://java.sun.com/j2se/downloads.html können sie die aktuelle JRE downloaden.

Kopieren Sie die Datei „TeToo.jar" auf ihre Festplatte. Das Programm wird durch einen Doppelklick auf das Programm-Icon gestartet.

Ich empfehle, eine Verknüpfung zu „TεToo" auf dem Desktop und/oder im Startmenü zu erstellen.

Beachten Sie, dass beim ersten Start des Programms eine Datei namens „options.dat" im Verzeichnis von TeToo.jar erstellt wird. In ihr werden die Programm-Einstellungen abgespeichert (siehe Abschnitt 5.1).

5. Das Hauptfenster

Nachdem Sie das Programm gestartet haben, öffnet sich das Haupt-Fenster mit dem Titel „Template-Tool“. Hinter dem Titel erscheint die Angabe des Benutzers. Außerdem sehen sie die Menüleiste mit den Punkten „File“ und „Project“. Unterhalb des Menüs erscheint eine freie graue Fläche auf der später die Projekt-Register angezeigt werden.
Unter dem Menüpunkt „File“ finden Sie Einträge um Informationen über das Programm zu erhalten („About Template-Tool“), die Einstellungen des Programms zu ändern („Options“) und um das Programm zu beenden („Close“).
Wenn Sie das Programm zum ersten Mal ausführen, sollten Sie die Programm-Einstellungen ihren Wünschen entsprechend anpassen.

5.1 Tool-Einstellungen

Die allgemeinen Einstellungen zum Programm können unter dem Menüpunkt „File - Options“ geändert werden.

In der Version 1.0 des Tools können folgende Einstellungen für das Programm gesetzt werden.
Root Project-Folder: Mit dieser Einstellung kann der Ordner festgelegt werden, in dem standardmäßig alle Projektordner angelegt und gesucht werden.
CSS-Style-Editor: In diesem Feld kann man eine Applikation angeben, die geöffnet werden soll, um CSS-Stylesheets, die zu einem Projekt gehören, zu bearbeiten.
JavaScript-Editor: Diese Angabe dient dazu, eine externe Applikation zu bestimmen, mit der die JavaScript-Dateien des Projektes bearbeitet werden.
Username: Hier sollten Sie ihren Benutzernamen angeben.

Abbildung 1: Tool-Options-Editor

Die Einstellungen des Programms werden in der Datei „options.dat“ gespeichert und beim Start des Programms aus dieser Datei ausgelesen. Kann das Programm die Datei nicht finden, so wird eine entsprechende Fehlermeldung angezeigt. In diesem Fall müssen Sie die Einstellungen noch einmal vornehmen.

5.2 Erstellen und Öffnen eines Projektes

Um mit dem Tool arbeiten zu können, müssen Sie ein neues Projekt anlegen oder ein bereits bestehendes Projekt öffnen. Die entsprechenden Funktionen finden sie unter dem Menüpunkt „Project".
Wenn Sie ein bestehendes Projekt öffnen möchten, müssen Sie die entsprechende Projektdatei mit Hilfe eines Dateiauswahl-Fensters (FileChooser) auswählen. Das Tool wertet die Projektdatei aus und öffnet die zum Projekt gehörenden XSL-Templates. Die gesammelten Daten werden in den Projektregistern angezeigt.
Wenn Sie unter „Project - New" ein neues Projekt anlegen möchten, erscheint der Projekt-Editor.

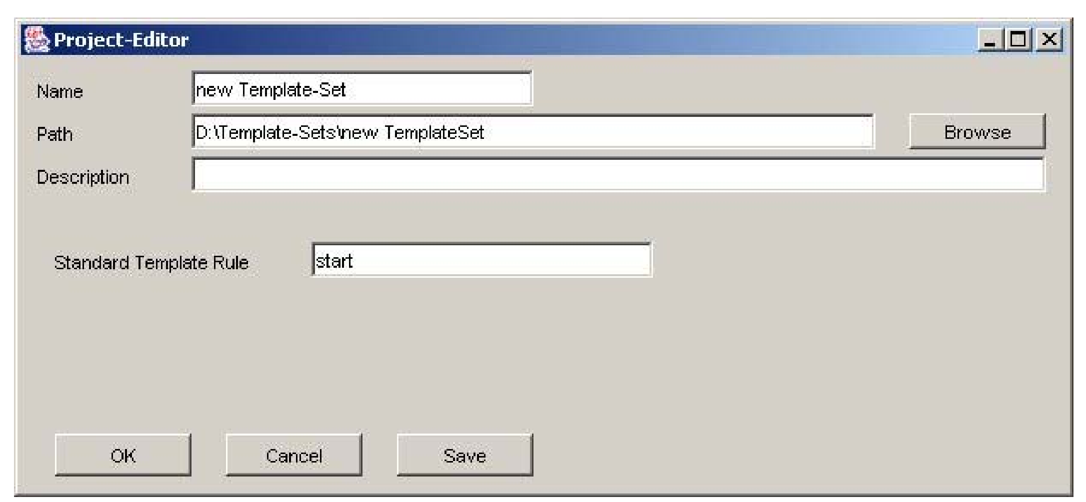


Abbildung 2: Project-Editor

Hier können Sie den neuen Namen des Projektes als auch den vorgesehenen Projektordner bestimmen. Wenn dieser Ordner noch nicht besteht wird das Programm nachfragen, ob er neu erzeugt werden soll. Des weiteren kann im Feld „Description" eine Projektbeschreibung angegeben werden.
Im Feld „Standard Template Rule" wird der Name der Template-Regel festgelegt, die in allen Start-Templates als erstes aufgerufen wird und für jedes Start-Template automatisch generiert wird.
Wenn Sie das neue Projekt abgespeichert haben und den Projekt-Editor verlassen, erscheinen im Hauptfenster die Projekt-Register.

Bei einem geöffneten Projekt, können Sie unter dem Menüpunkt „Project - Edit Properties" ebenfalls den Projekt-Editor öffnen. In diesem Fall ist es jedoch nur möglich die Beschreibung und die Standard-Template-Regel zu ändern. Außerdem steht noch ein Button zur Verfügung, mit dem Sie das gesamte Projekt löschen können.

Sobald Sie ein Projekt geöffnet oder erstellt haben, wird die entsprechende Projektdatei gesperrt. Dies hat das Ziel, eine mehrfache gleichzeitige Bearbeitung des Projektes zu verhindern. Die Sperre wird aufgehoben, wenn Sie das Projekt wieder schließen.
Wenn Sie versuchen, ein gesperrtes Projekt zu öffnen, erscheint eine Meldung, die angibt welcher User das Projekt augenblicklich bearbeitet und seit wann das Projekt gesperrt ist.

- 11 -

Wenn Sie sicher sind, dass eine gesperrte Datei von keinem anderen Benutzer bearbeitet wird, müssen Sie die Sperre manuell entfernen. Dazu müssen Sie im Projektordner die Datei „lock.ttl“ suchen und löschen.

5.3 Die Projekt-Register

Die Projekt-Register sind sichtbar, wenn ein Projekt geöffnet ist. Über die Register werden alle Informationen über das Template-Set angezeigt und können geändert werden. Oberhalb der Projektregister werden zusätzlich der Projektname und der Pfad des Projektordners angezeigt.

5.3.1 Parameter

Beim ersten angezeigten Register handelt es sich um das Parameter-Register. In diesem Register werden die Importangaben und die Parameter für die Start-Templates (Standard- und Spezial-Templates) angezeigt.

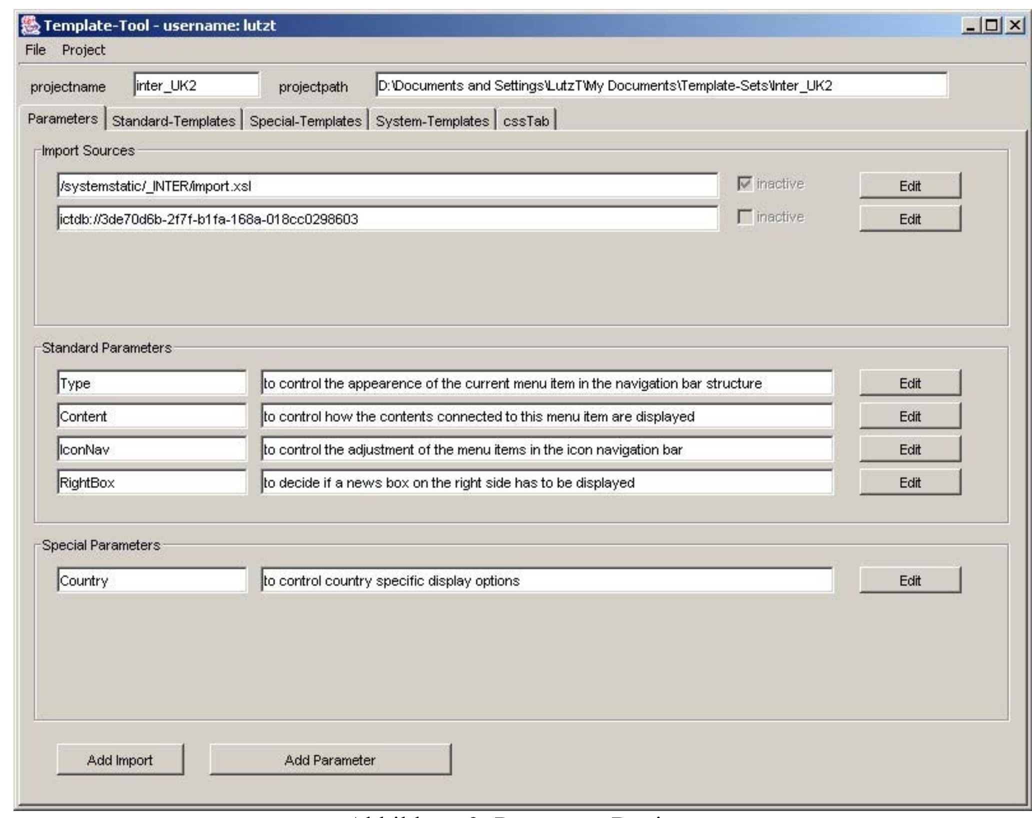


Abbildung 3: Parameter-Register

Mit Hilfe der „Edit“-Buttons lassen sich Editoren zum Bearbeiten der Imports (siehe Abschnitt 6: Import-Editor) und Parameter (siehe Abschnitt 7: Parameter-Editor) aufrufen. Die Buttons zum Hinzufügen neuer Imports und Parameter in der linken unteren Ecke rufen ebenfalls diese Editoren auf.

Alle Änderungen, die in diesem Register vorgenommen werden, wirken sich auf alle Standard- und Spezial-Templates aus.

- 12 -

5.3.2 Template-Register

Es gibt drei Register zur Anzeige der Templates, unterteilt in „Standard-Templates“, „Spezial-Templates“ und „System-Templates“. Angezeigt wird jeweils der Name, sowie eine Kurzbeschreibung des Templates. Über einen „Edit“-Button kann für jedes Template der Template-Editor (siehe Abschnitt 8) geöffnet werden.

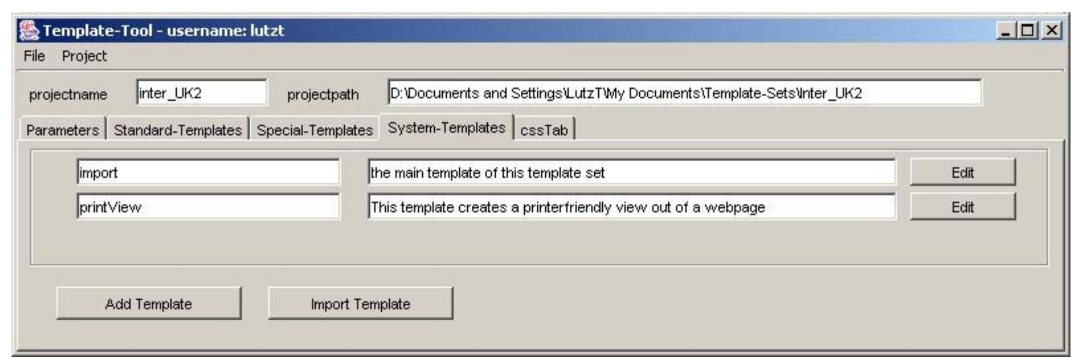


Abbildung 4 Template-Register

In jedem Register gibt es einen Button zum Hinzufügen eines neuen Templates, welches dann im Template-Editor bearbeitet wird.
Bei Spezial- und System-Templates gibt es zusätzlich noch die Möglichkeit, bereits existierende XSL-Dateien zu importieren. Mit Hilfe eines FileChoosers wird die zu importierende Datei ausgewählt. Anschließend wird die XSL-Datei geparst und nach den Regeln des Template-Tools in ein neues Template umgewandelt, das im gleichzeitig geöffneten Editor noch bearbeitet werden kann. Da sich die Standard-Templates nur in den Werten der Parameter unterscheiden, ist eine Import-Funktion an dieser Stelle nicht sinnvoll.

Durch das Importieren eines Templates können Daten aus der XSL-Datei verloren gehen. Das ist der Fall, wenn die XSL-Datei von der vom Programm erwarteten Syntax abweicht. Die im Parameter-Register definierten Parameter werden in importierte Special-Templates integriert. Vor dem Abspeichern des importierten Templates, müssen diesen Parametern zulässige Werte zugewiesen werden.

5.3.3 CSS / Javascript

Im letzten Register werden die zum Projekt gehörenden CSS-Stylesheets und JavaScript-Dateien angezeigt. Für beide Datei-Typen besteht die Möglichkeit die bestehenden Dateien in einem Editor zu bearbeiten, bzw. neue zu importieren. Dateien die importiert werden sollen, werden in einem FileChooser ausgewählt und in den Projektordner kopiert.

6. Import-Editor

Der Import-Editor wird geöffnet, wenn im Parameter-Register des Hauptfensters der „Edit“-Button hinter einer Importanzeige betätigt wird oder ein neuer Import hinzugefügt werden soll.
Im Textfeld „Source“ wird die Importquelle eingetragen. Zusätzlich kann noch festgelegt werden, ob die Importquelle aktiv sein soll oder nicht. Wird die Quelle als inaktiv definiert, wird sie im Template-Code auskommentiert.
Mit den Buttons „OK“ und „Save“ werden die Änderungen übernommen. „Cancel“ beendet den Editor ohne Übernahme der Änderungen. Über den Button „Delete“ kann die Importquelle aus dem Projekt entfernt werden.

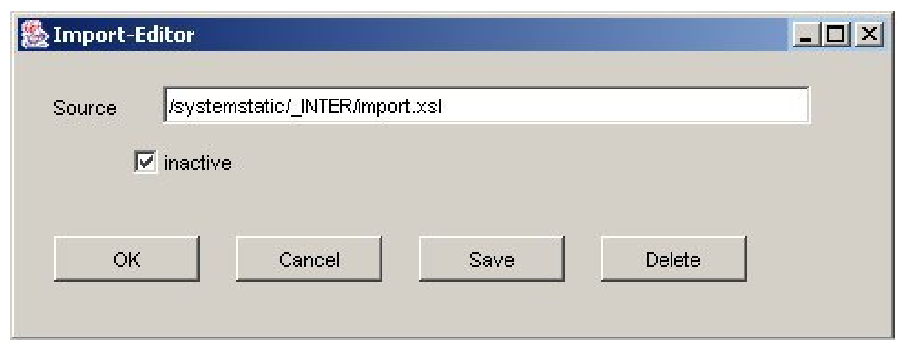


Abbildung 5: Import-Editor

7. Parameter-Editor

Mit dem Parameter-Editor werden die Einstellung für Parameter vorgenommen. Dazu gehören Parametername, Parameterbeschreibung, sowie die Angabe zulässiger Werte.
Die zulässigen Werte werden durch Kommata getrennt. Beginnt die Eingabe mit einem Komma, oder folgen zwei Komma aufeinander, wird der Wert *no value* in die Liste der erlaubten Werte hinzugefügt, das heißt für diesen Parameter muss kein Wert angegeben werden. Wenn das Feld „Possible values“ leer ist, kann der Parameter beliebige Werte (*any value*) bekommen.
Es gibt die Möglichkeit einem Parameter in allen Start-Templates den gleichen Wert zu geben. Dazu muss die Check-Box „value for all start-templates“ markiert sein. Der Wert muss im folgenden Textfeld eingetragen werden.

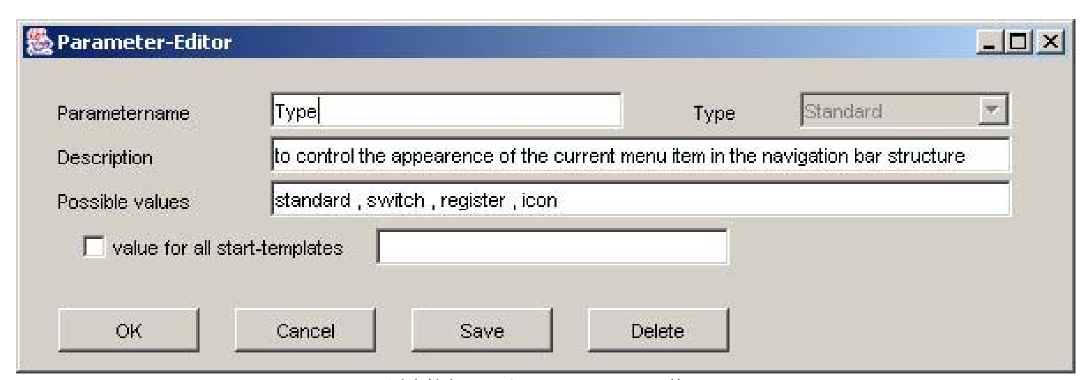


Abbildung 6: Parameter-Editor

Wird ein Parameter neu angelegt, kann man im Feld „Type“ zwischen Standard- und Spezial-Parameter wählen.

- 14 -

8. Template-Editor

Neben dem Hauptfenster bildet der Template-Editor das Kernstück des Tools. Er dient zum Bearbeiten eines Templates. Um die Informationen übersichtlicher zu gestalten, gibt es beim Template-Editor verschiedene Register. Die Anzeige der Register ist vom Typ des Templates abhängig.

8.1 Allgemeine Informationen

Dieses Register enthält die allgemeinen Angaben über das ausgewählte Template. Dazu gehören der Templatename, Version und eine Beschreibung, die in den entsprechenden Feldern geändert werden können. Die Werte für das Web (Projekt) und den Template-Typ lassen sich hier nicht ändern.
Der Button „Delete“ löscht das Template aus dem Projekt und ggf. auch die entsprechende XSL-Datei von der Festplatte.
Die „show code“-Funktion öffnet ein Fenster, in dem der gesamte XSL-Code des Templates betrachtet werden kann. Er kann nicht in diesem Fenster bearbeitet werden.

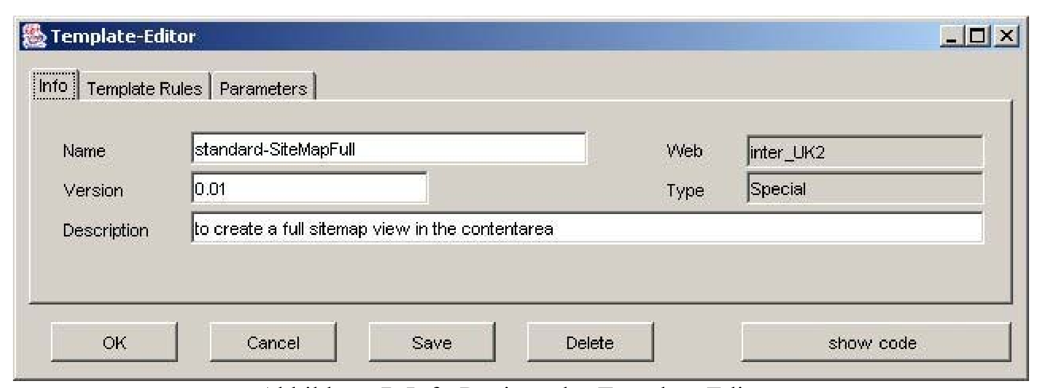


Abbildung 7: Info-Register des Template-Editors

8.2 Template Rules

Im Register “Template Rules” erhält man eine Übersicht über alle im Template angelegten Template-Regeln. Neben dem Templatenamen erhält man eine kurze Beschreibung.

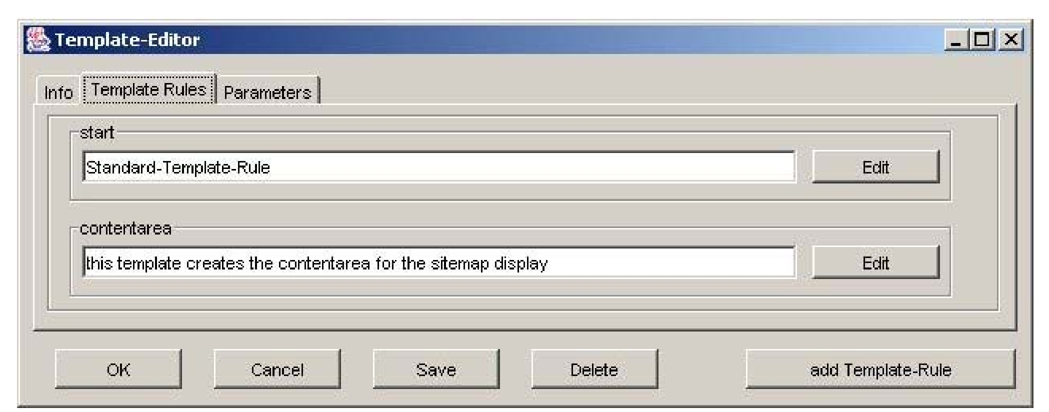


Abbildung 8: „Template Rules“-Register des Template-Editors

Über einen „Edit“-Button kann man den TemplateRule-Editor zum Bearbeiten einer Template-Rule öffnen. Der Button „Add Template-Rule” öffnet den TemplateRule-Editor zum Erstellen einer neuen Template-Regel.

8.2.1 TemplateRule-Editor

Der TemplateRule-Editor dient zum Bearbeiten einzelner Template-Regeln. In den Textfeldern im oberen Bereich können Name und Beschreibung der Template-Regel geändert werden. Zusätzlich kann ein „Match“-Knoten eingetragen werden, auf dem diese Template-Regel ausgeführt wird. Ist dieses Feld leer, kann die Template-Regel nur durch einen Aufruf aus einer anderen Template-Regel ausgeführt werden.

Im Textfeld „Rule-Code“ wird der Programmcode für die Regel erstellt. Bevor eine Regel gespeichert werden kann, wird sie auf XML-Konformität geprüft. Ist die XML-Konformität nicht gegeben, wird die Regel nicht gespeichert und der Editor kann nicht beendet werden. Es wird eine entsprechende Fehlermeldung angezeigt.

Eine Änderung der Standard-Template-Regel (definiert im Projekt-Editor) bezieht sich auf alle Standard- und Spezial-Templates und ändert bei all diesen Templates diese Template-Regel.

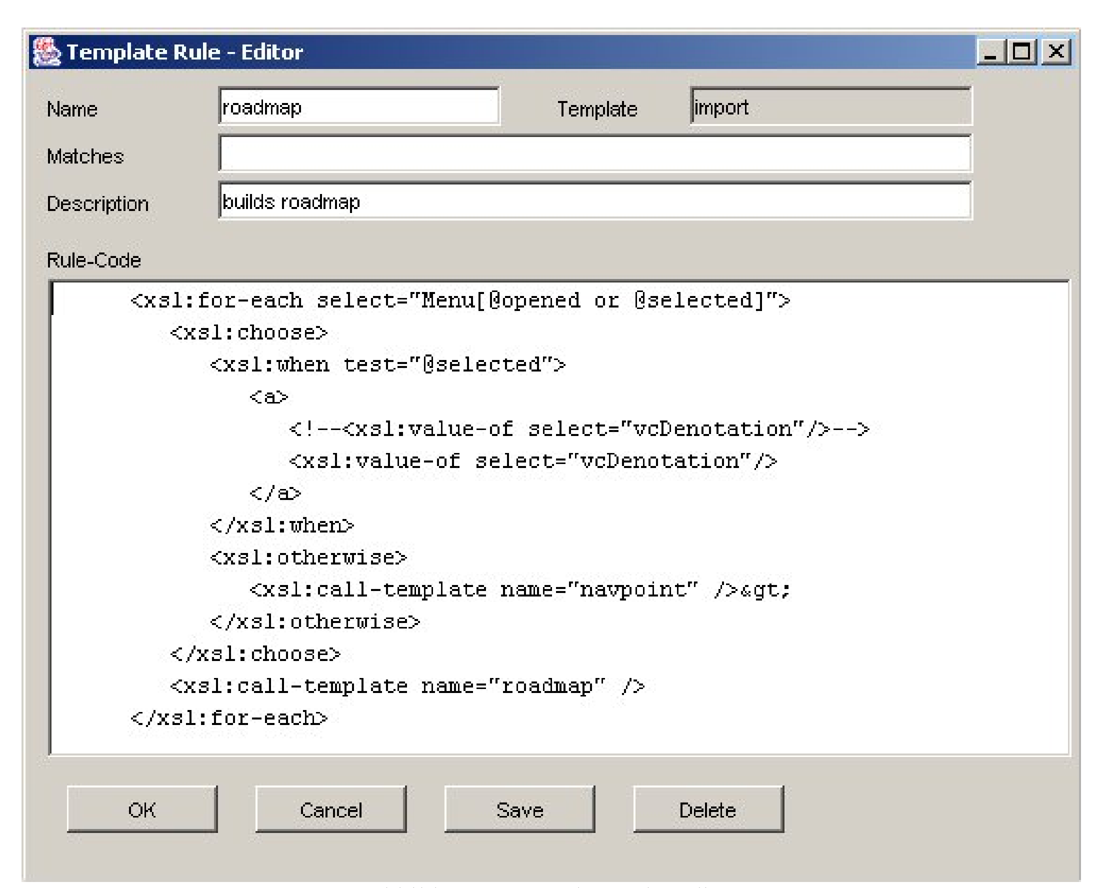


Abbildung 9: TemplateRule-Editor

- 16 -

8.3 Parameter

Das Register "Parameters" ist nur für Standard- und Spezial-Templates verfügbar. In dem Feld „Value" können die Werte für die im Parameter-Register des Hauptfensters definierten Parameter festgelegt werden. Die wichtigsten Eigenschaften der Parameter werden angezeigt.
Ist für einen Parameter die „value for all"-Funktion (Hauptfenster) aktiviert, kann der Wert nicht geändert werden.

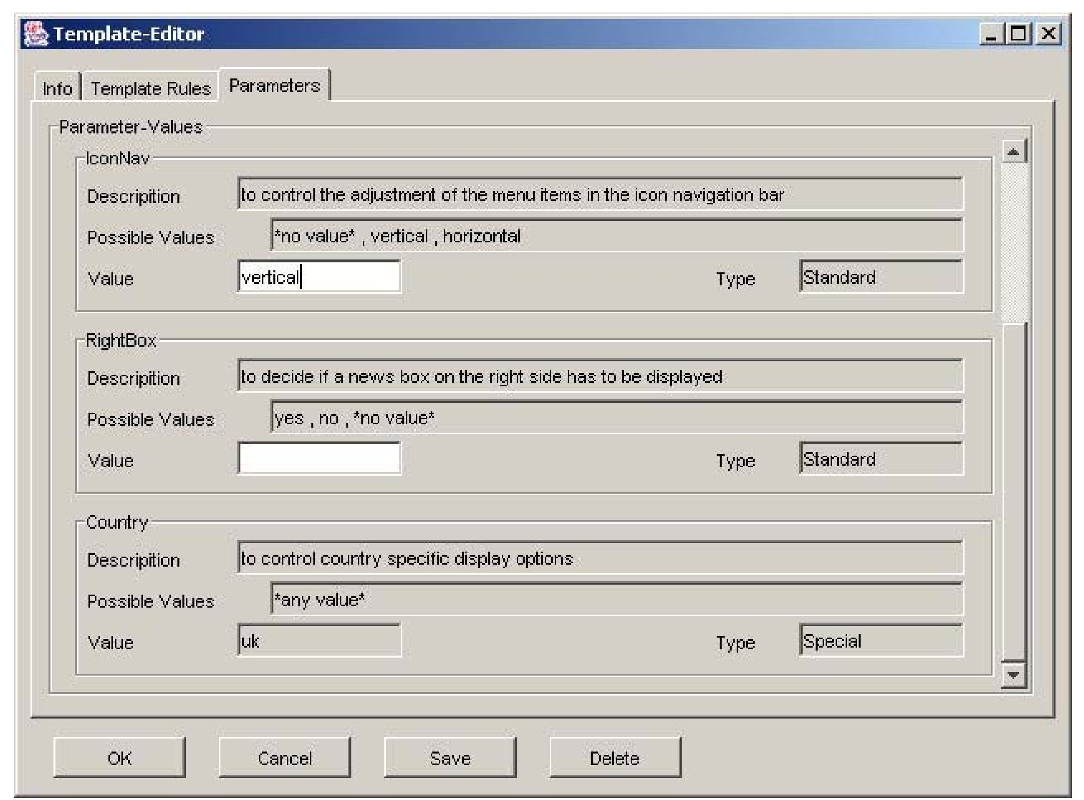


Abbildung 10: Parameter-Register des Template-Editors

8.4 System-Parameter

Dieses Register wird nur bei System-Templates angezeigt. Für das ausgewählte Template können spezielle Parameter definiert werden. Es ist möglich, als Parameter-Wert XSL-Code anzugeben.

Wie bei den Template-Regeln ist es auch bei den System-Parametern möglich, mit Hilfe des „Edit"-Buttons die System-Parameter zu bearbeiten und über einen speziellen Button neue System-Parameter zu erzeugen.

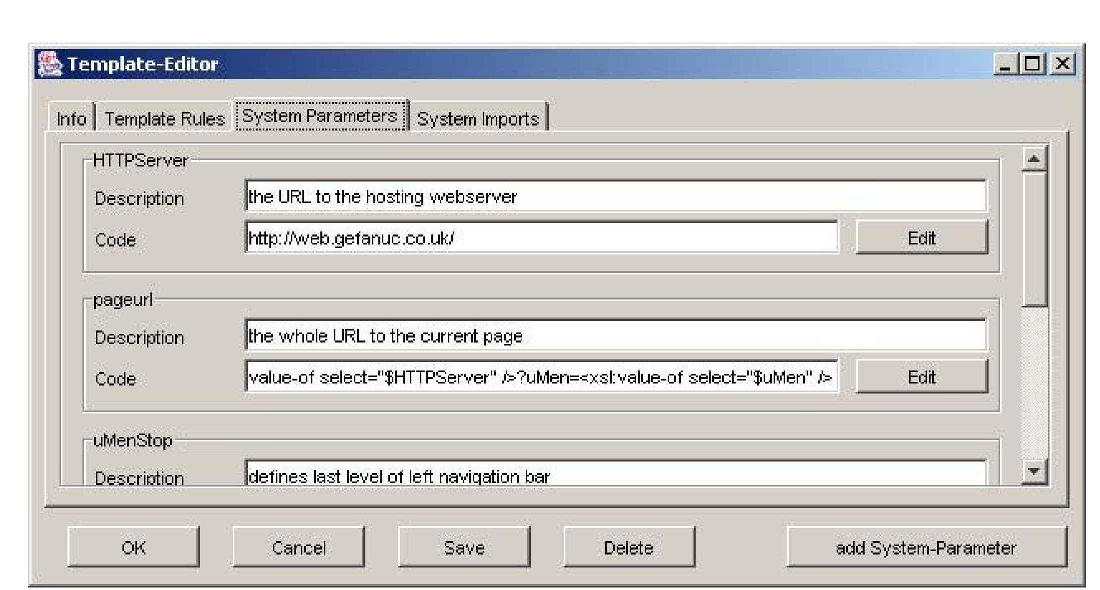


Abbildung 11: System-Parameter-Register des Template-Editors

8.4.1 System-Parameter editieren

Zum Bearbeiten der System-Parameter gibt es ebenfalls einen Editor. Funktion und Handhabung ist der des TemplateRule-Editors ähnlich. Einziger Unterschied ist das fehlende Eingabefeld für den „Match"-Knoten.

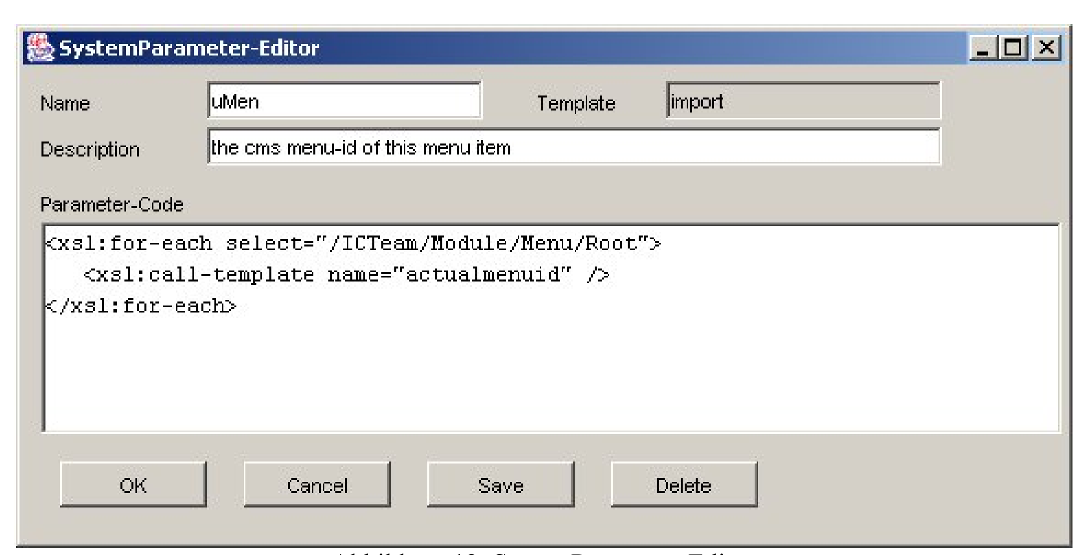


Abbildung 12: SystemParameter-Editor

8.5 System-Importe

Ebenfalls nur bei System-Templates sichtbar ist das Register für System-Importe. Hier können individuell Importquellen für das ausgewählte System-Template festgelegt werden. Eine Änderung kann im entsprechenden Textfeld vorgenommen werden. Ist ein Feld leer, wird die Importquelle entfernt. Ein Button erlaubt das Hinzufügen neuer Importangaben.

9. CSS-Editor und JavaScript-Editor

Die Editoren für CSS-Stylesheets und JavaScript-Files bieten die Möglichkeit, die Dateien umzubenennen oder aus dem Projekt zu löschen.
Über den Button „Edit Styles“ bzw. „Edit JavaScript“ wird die Datei mit den in den Tool-Einstellungen festgelegten externen Programmen zum Bearbeiten geöffnet.

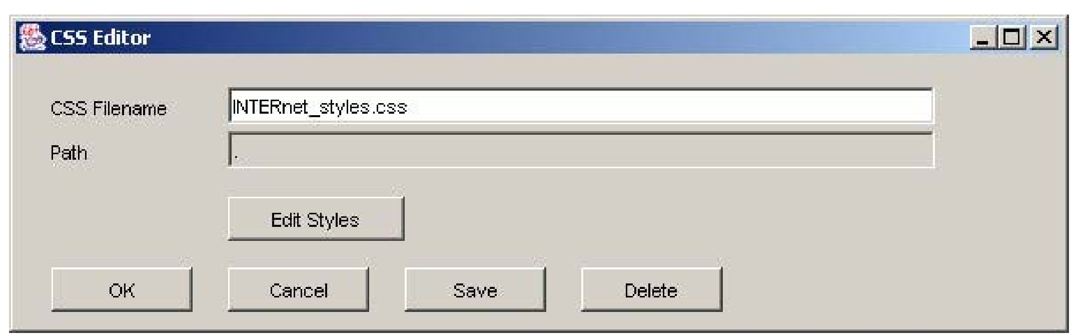


Abbildung 13: CSS-Editor

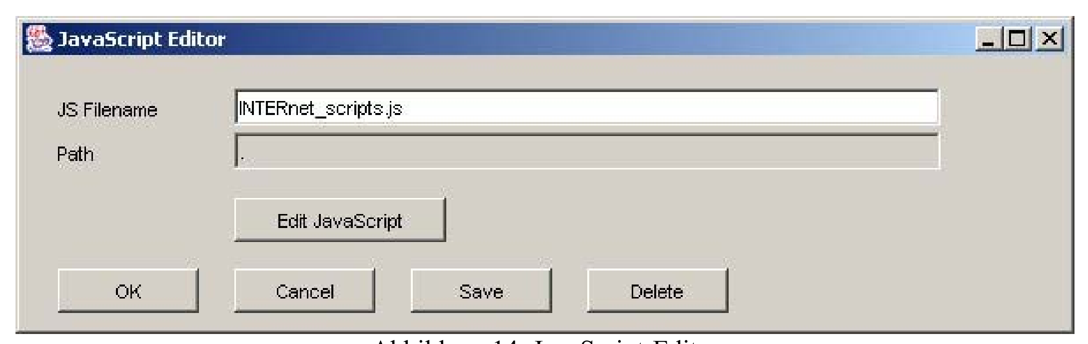


Abbildung 14: JavaScript-Editor

7.7 Screenshots von Intra-, Extra- und Internet

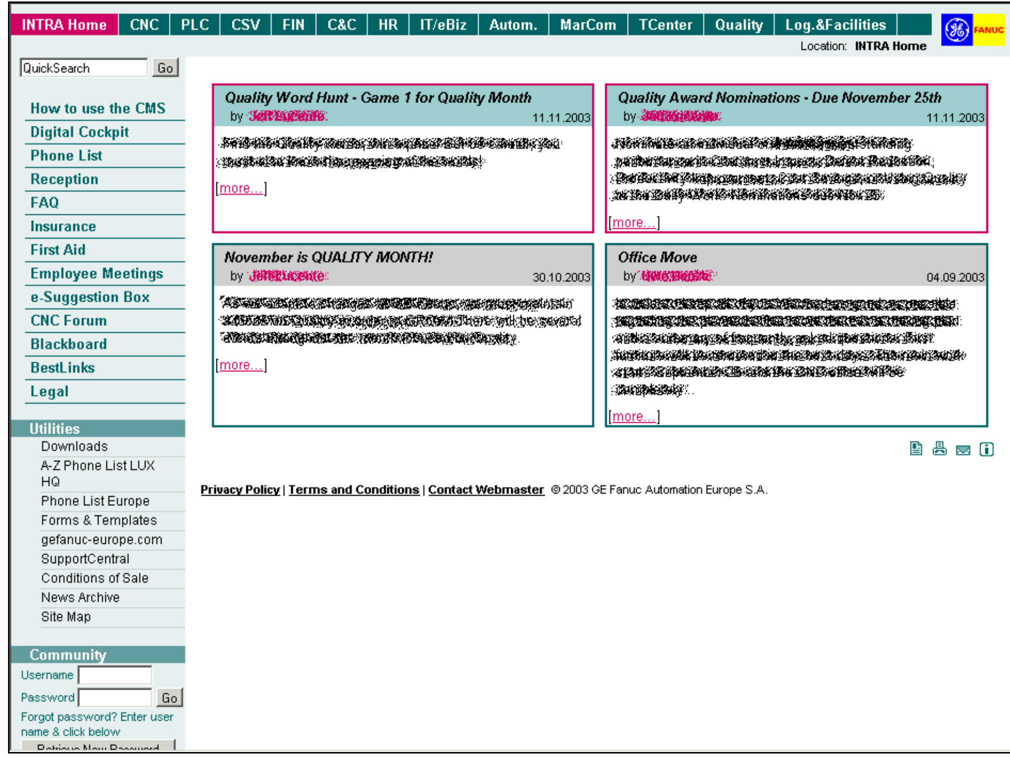


Abbildung 7-14: Intranet-Startseite (News Darstellung)
Quelle: GE Fanuc Intranet

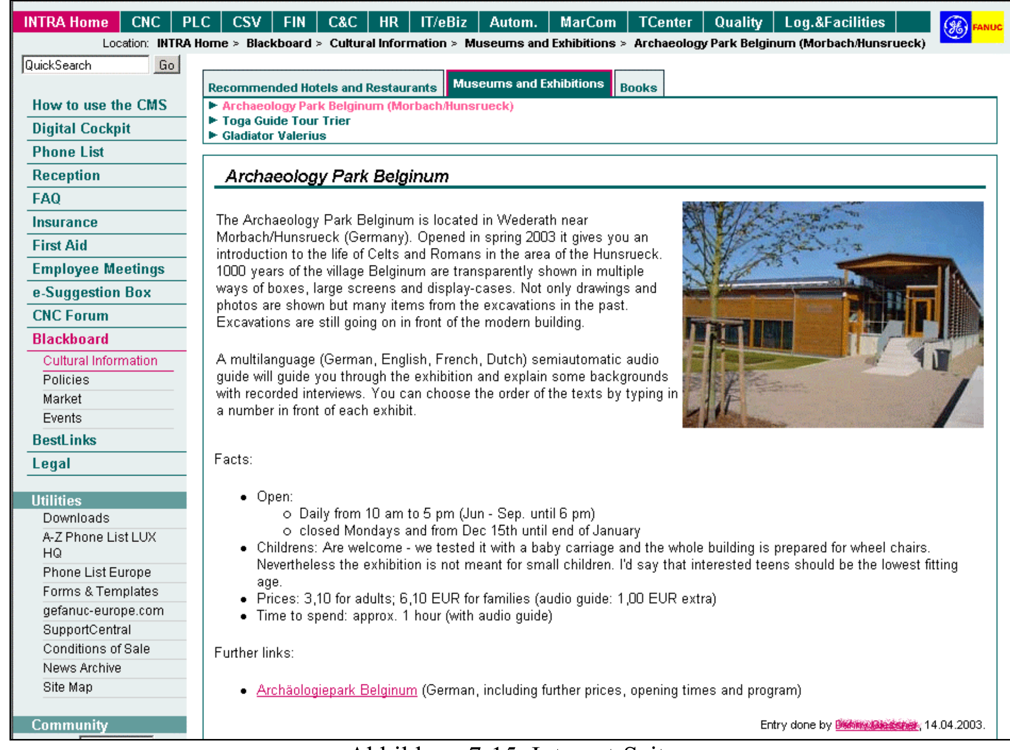


Abbildung 7-15: Intranet-Seite
Quelle: GE Fanuc Intranet

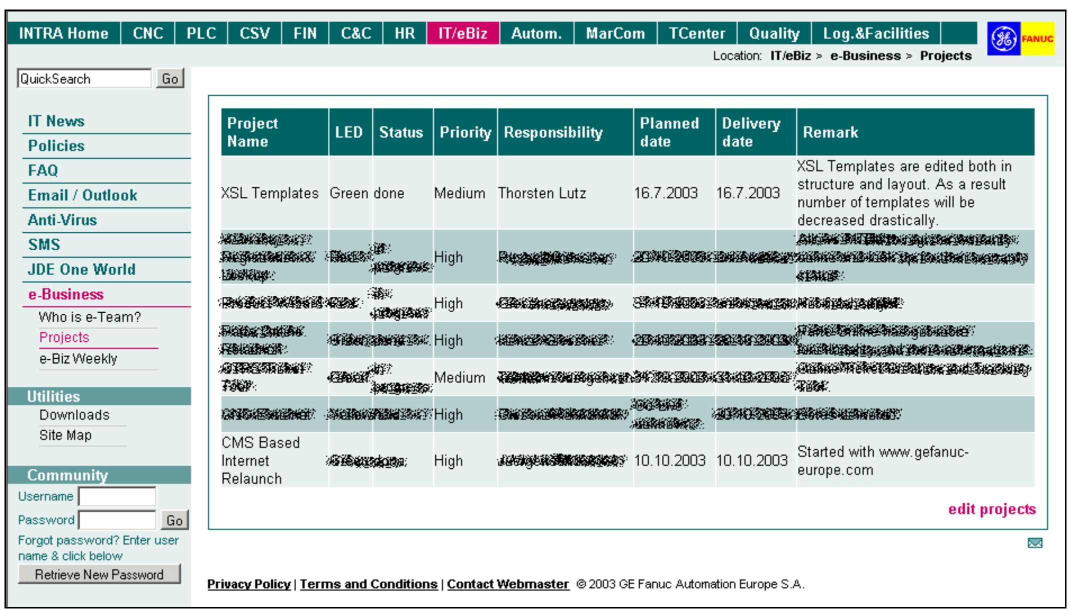


Abbildung 7-16: Intranet-Seite (Darstellung als Attribut-Tabelle)
Quelle: GE Fanuc Intranet

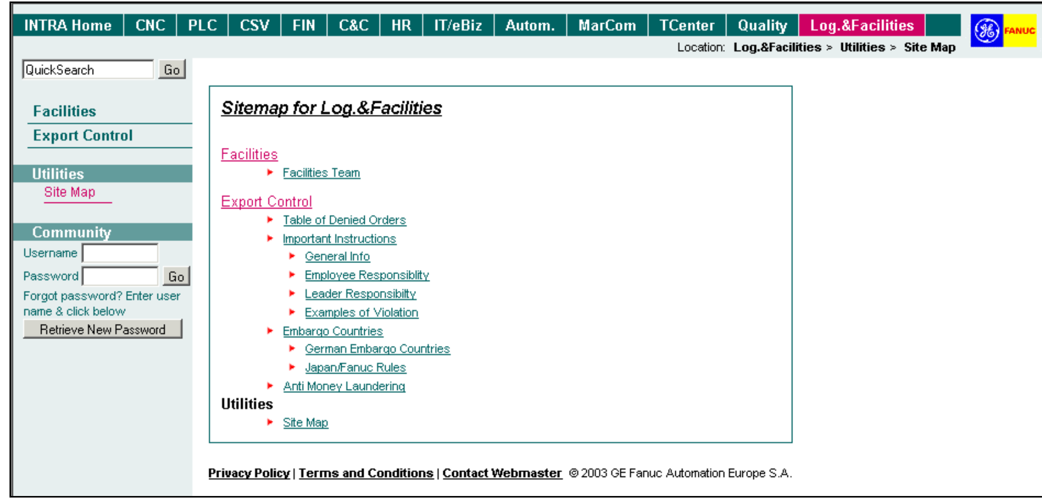


Abbildung 7-17: Intranet-SiteMap
Quelle: GE Fanuc Intranet

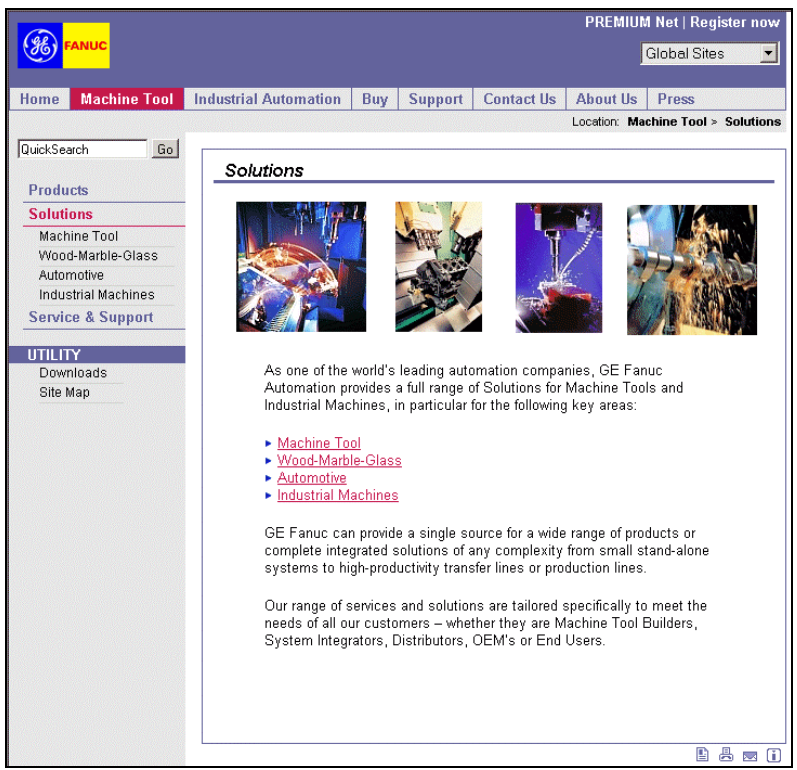


Abbildung 7-18: Internet-Seite
Quelle: http://www.gefanuc-europe.com/cmsinter/broker.jsp?uMen=7ce70515-fe54-45f7-a51b-018cc0298603 (13.11.2003)

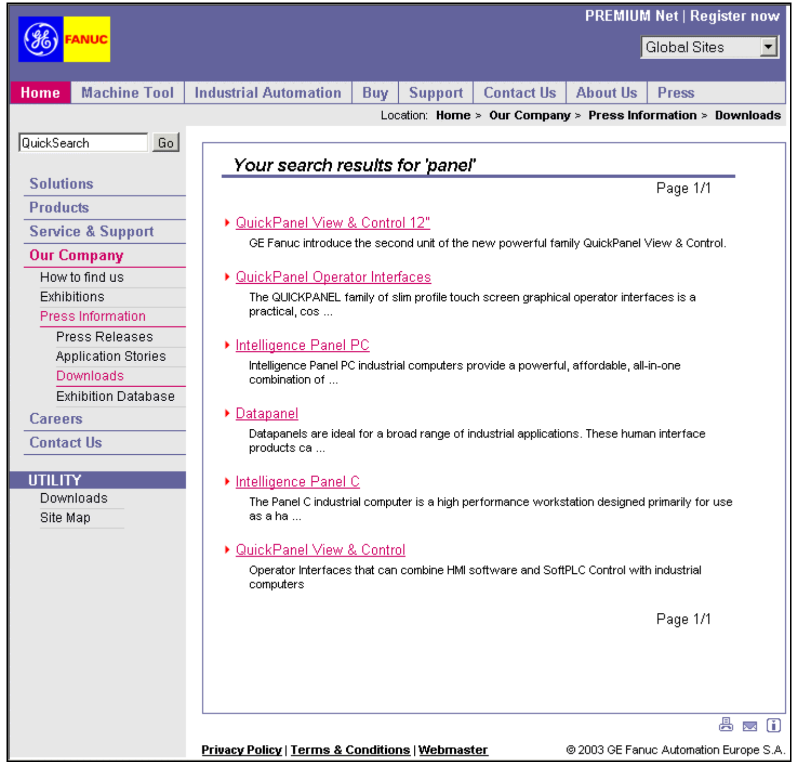


Abbildung 7-19: Internet-Seite (Suchergebnis)
Quelle: http://www.gefanuc-europe.com/cmsinter/broker?uMen=7ce70515-fe54-45f7-a51b-018cc0298603&uTem=5ef7107f-ee3a-6bea-1bcf-199656786c1c (13.11.2003)

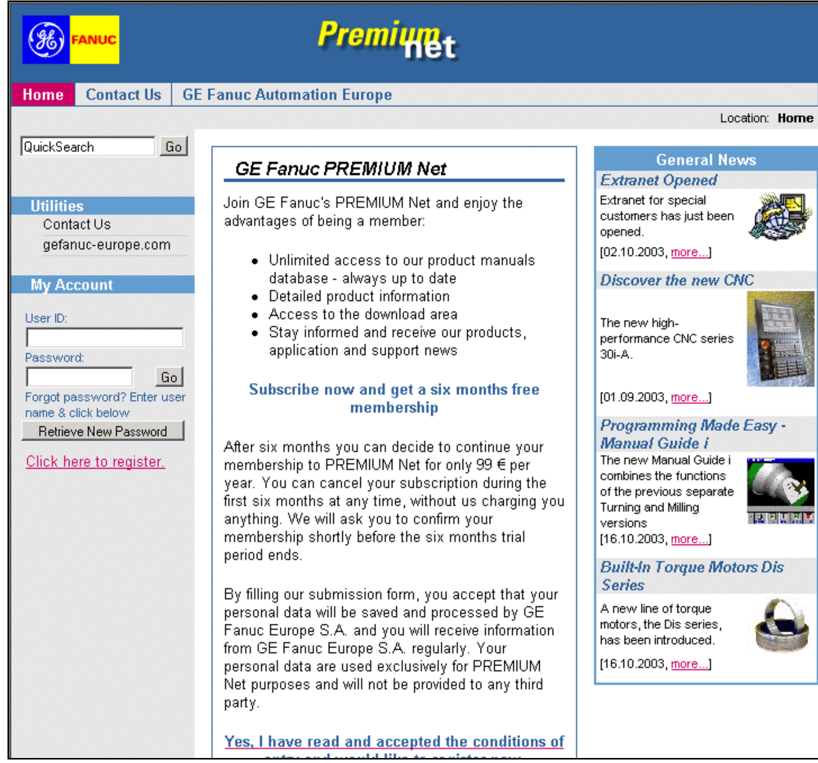


Abbildung 7-20: Extranet-Startseite
Quelle: GE Fanuc Extranet Testsystem

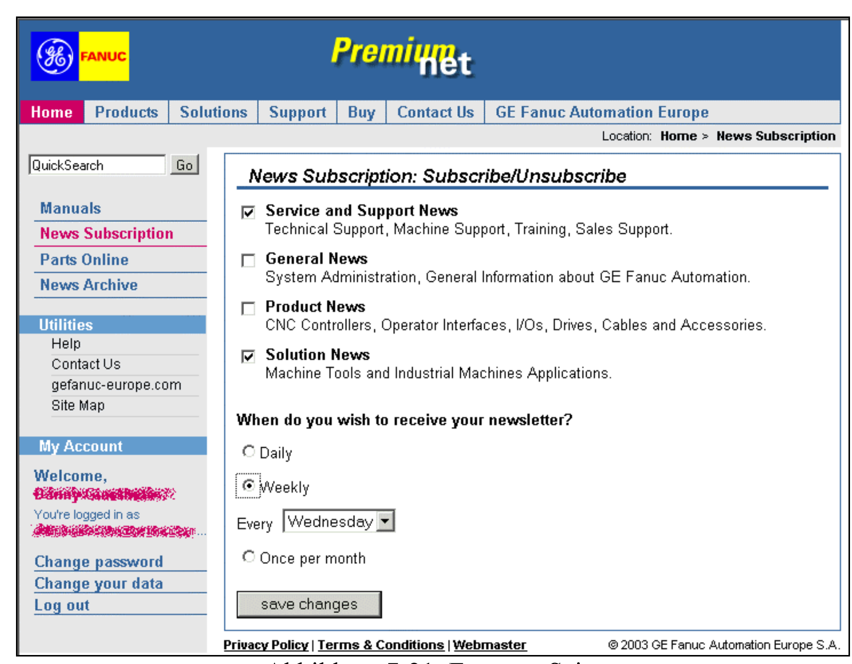


Abbildung 7-21: Extranet-Seite
Quelle: GE Fanuc Extranet Testsystem

8 Literaturverzeichnis

Büchner, Traub, Zahradka, Zschau; Web Content Management;
Galileo Press GmbH Bonn, 1. Auflage 2001;
ISBN 3-934358-86-1

Dellwig Elmar; easy JavaScript; Markt+Technik Verlag, 2001
ISBN 3-8272-6031-0

Die deutsche Content Management Site; Online im Internet;
URL: http://www.contentmanager.de
(13.11.2003)

Google Groups - comp.lang.java; Online im Internet;
URL: http://groups.google.de/groups?hl=de&lr=&ie=UTF-8&group=comp.lang.java (August 2003)

Google Groups - comp.lang.javascript; Online im Internet;
URL: http://groups.google.de/groups?hl=de&lr=&ie=UTF-8&group=comp.lang.javascript (Juni/Juli 2003)

Herpers F.J., Sebestyen T.J.; Das Einsteigerseminar XSL;
Verlag moderne industrie Buch AG & Co. KG, Landsberg; 2002;
ISBN 3-8266-7209-7

IC Team Trier GmbH; Benutzerhandbuch ICContent Content Management System; IC Team GmbH Trier, 2001

IC Team Trier GmbH; Online im Internet; URL: http://www.icteam.de/
(13.11.2003)

Münz Stefan; SelfHTML - HTML-Dateien selbst erstellen; Online im Internet;
URL: http://selfhtml.teamone.de/ (13.11.2003)

o.V.; The Finest In Stylesheets; Online im Internet;
URL: http://www.css4you.de/ (13.11.2003)

o.V.; Universität Paderborn – Didaktik der Informatik; Geschichte des Hypertextes; Online im Internet; URL: http://ddi.uni-paderborn.de/didaktik/lehre/sose2000/tide/html_geschichte.html
(13.11.2003)

Scholz Michael; Web-Content-Management-Basiscs 1; Online im Internet;
URL: http://www.content-syndication-broker.de/crm-wcms-artikel-seite31-folge3.html (13.11.2003)

Seeboerger-Weichselbaum M.; Das Einsteigerseminar XML;
Verlag moderne industrie Buch AG & Co. KG, Landsberg; 2002;
ISBN 3-8266-7201-1

Sun Microsystems; Java 2 Platform, Standard Edition, v 1.4.2 API Specification; Online im Internet; URL: http://java.sun.com/j2se/1.4.2/docs/api/index.html (13.11.2003)

World Wide Web Consortium; Online im Internet; URL: http://www.w3.org/ (13.11.2003)